
LOAD
REVISTA PARA USUARIOS DE
MSX
AÑO 1 Nº 3 ₳ 2.30 REP. ARGENTINA
TALENT
MSX DPC-200
VS. EL RESTO
PROGRAMAS
• CAJERO
• EDITOR ASSEMBLER
• JUEGO DE LA OCA
REVISION DE
SOFTWARE
ALTA
RESOLUCION
SPRITES
VIVOS



AÑO 1 Nº 3



Director General
Ernesto del Castillo

Director Editorial
Cristian Pusso

Director Periodístico
Fernando Flores

Director Financiero
Javier Campos Malbrán

Arte y Diagramación
Fernando Amengual
Tamara Migelson

Coordinador
Ariel Testori

Redacción
Eduardo Mombello
Andrea Sabin Paz

Departamento de Avisos
Oscar Devoto

Departamento de Publicidad
Guillermo González Aldalur

Departamento fotográfico
Víctor Grubicy

Load Revista para usuarios de MSX es una publicación mensual editada por Editorial PROEDI S.A., Paraná 720, 5º Piso, (1017) Buenos Aires. Tel.: 46-2886 y 49-7130. Registro Nacional de la Propiedad Intelectual: E.T. M. Registrada. Queda hecho el depósito que indica la Ley 11.723 de Propiedad Intelectual.

Impresión: Calcotam. Fotocromo tapa: Columbia. Fotocomposición: Van Waveren.
Los ejemplares atrasados se venderán al precio del último número en circulación.

Distribuidor en Capital: Martino, Juan de Garay 358, P.B. Capital.
Distribuidor interior: DGP: Hipólito Yrigoyen 1450, Capital Federal. T.E. 38-9266/9800.

Sumario

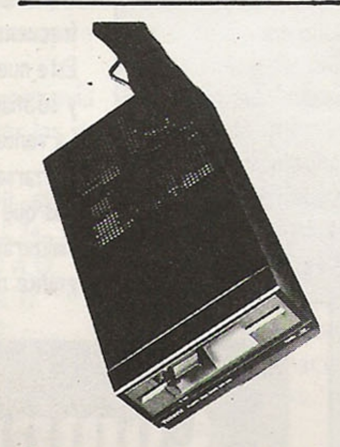
SPRITES VIVOS

Abramos bien los ojos y aceitemos nuestra mente porque aquí va la palabra mágica que hará posible todo este desarrollo: Hook! (pág. 5)

¿QUE ES LA VRAM?

Para aprovechar al máximo las posibilidades del Basic, es muy importante no programar siempre con las mismas instrucciones. (pág. 6)

DOS



Comenzaremos viendo algunas funciones complementarias a los comandos disponibles de este sistema operativo, desde Basic. (pág. 10)

TALENT MSX DPC-200 VS. EL RESTO

Es un cuadro de las especiales características de la primera máquina lanzada por TALENT, y el resto de las computadoras. (pág. 18)

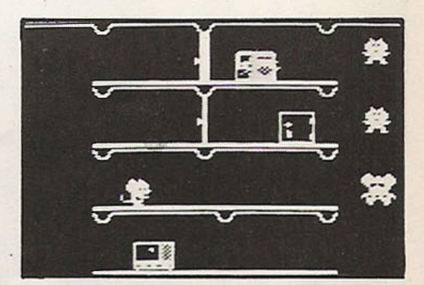
MSX LOGO

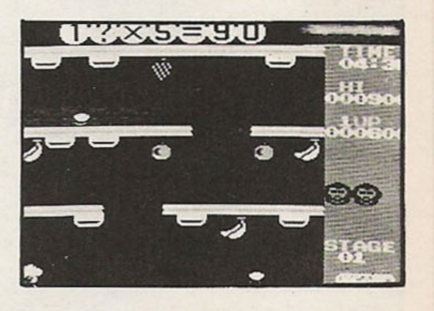
Otra vez la empresa TELEMATICA extiende sus desarrollos a las manos de los usuarios de sus computadoras. Se trata esta vez del programa que nos permitirá trabajar en lenguaje LOGO. Pero esta vez no viene solo, lo acompaña un manual preparado por Hilario Fernandez Long y Horacio Reggini. (pág. 16)

ALTA RESOLUCION

Aprendamos a aplicar todas las facilidades gráficas que tiene nuestra computadora y veamos algunos ajemplos. (pág. 26)

SECCIONES FIJAS





PROGRAMAS

Industria Argentina

La producción nacional de computadoras personales ya es una realidad indiscutible, como lo podemos observar con la Talent MSX.
La mayor parte del proceso de fabricación está hecho en Argentina, con componentes comprados aquí. Entre ellos podemos citar: circuitos impresos, resistores, capacitores, transformadores, algunos conectores , fuentes de poder, piezas plásticas, disipadores, tornillería, cables, manuales y embalajes.
Los procesos locales son la inserción y soldado de componentes en las plaquetas de circuito impreso, así como el armado final y pruebas.
La fabricación de las Talent MSX se realiza en una planta en la ciudad de San Luis que mantiene un ritmo de fabricación de 40.000 computadoras por año y 2.500 periféricos.
Además, las perspectivas son promisorias: espera duplicar estas cifras en breve tiempo.

LOS EDITORES



TAMBIEN SONY ES MSX

Esta compañía japonesa mostró su modelo MSX-2. Una ampliación en todo sentido acompaña a estas computadoras, desde una ROM de 64K y una RAM de 192K, de los cuales 128K son destinados para realizar gráficos.

IBM HACIA MSX

Como se pudo conocer en INFORMAT 86, IBM lanzará pronto los nuevos modelos de computadoras que incorporan el sistema MSX.
Esto no significa que los nuevos ordenadores sean totalmente MSX, pero sí se sabe que habrá compatibilidad sobre el software de gestión y aplicaciones.
No hubo demasiada información de este tema, sobre todo de la parte técnica que formarán a estos ordenadores.
Se marca, con esto, el avance del nuevo sistema, que se expande rápidamente, y ha seducido a una de las marcas más importantes del mundo de la informática.



UN TEACH-ROBOT DE VIETA

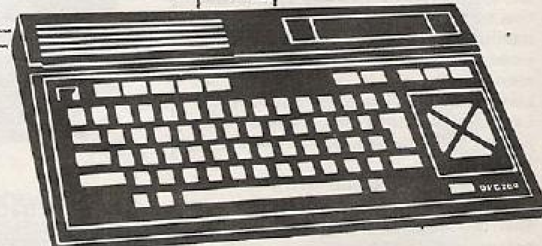
La firma VIETA —esta compañía es más conocida por su comercialización de los equipos Pioneer— ha creado un novedoso brazo-robot que ayudará a enseñar y resolver los problemas de robótica más frecuentes e importantes.
Este nuevo periférico es compatible con las MSX y su manejo se realiza desde el software.
La venta de este kit permite a los usuarios familiarizarse con este tipo de monstruos electrónicos que suponemos en poco tiempo se comercializarán de tal forma que pasarán a ser un integrante más de cada casa.

UN MOUSE PARA MSX

Parece que la compañía TELEMATICA está decidida a informatizarnos a todos y con el mejor nivel posible.
Cada vez nuestra Talent se parece más a una PC. Al borde del infarto programático, nos puso la noticia que llegó hasta nuestros oídos, y la presencia de este ratón electrónico que no tiene absolutamente nada que envidiarle a los de otras maquinolas.
Este pequeño ser, que pesa 200 gramos con cola y todo, viene acompañado por un programa que nos permitirá diseñar, almacenar e imprimir todas las pantallas que nuestra imaginación nos sugiera.
Además, puede ser usado simplemente como joystick y mil utilidades más que veremos en una próxima nota dedicada a este hermoso roedor de pantallas.

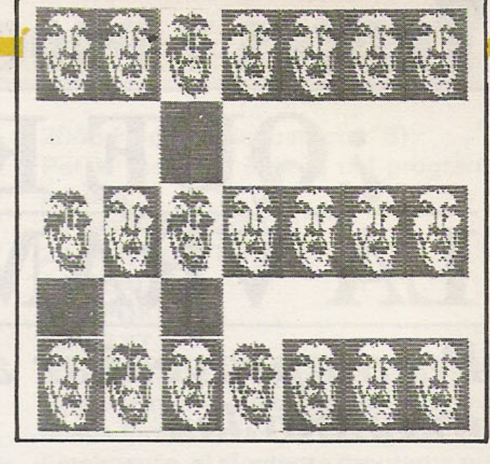






SPRITES VIVOS

Abramos bien los ojos y aceitemos nuestra mente, porque aquí va la palabra mágica que hará posible todo este desarrollo: Hook!





Vamos a mostrar la forma de darles movimiento automático a nuestros SPRITES. Esto quiere decir que los sprites van a circular por la pantalla a la velocidad y con la dirección que uno quiera, sin tener que hacer una infinidad de PUT SPRITES. El encargado de este trabajo va a ser un programita en Assembler. Pero a no desesperar pues aunque no sepamos Assembler, este método de movimiento automático nos resultará muy sencillo.

Las primeras líneas de todo programa que utilice este sistema deben ser las siguientes:

```
10 CLEAR 200, -6016                 LD HL, 6912
20 FOR A = -6016 TO -6005           CALL 74
30 READ A $                         ADD A,0
40 POKE A, VAL ("&H" + A$)          CALL 77
60 DATA 21,0,1B,CD,4A,0,C6,0,CD,4D,0
70 DATA C9
```

¿Qué hicimos? Muy fácil: cargamos el programa Assembler, que está en las líneas DATA, en un lugar de la memoria en que no "moleste". Para eso, en la línea 10 el CLEAR reserva ese lugar de la memoria, para que nuestro programa no pueda ser borrado. El programa es cargado desde la posición de memoria –6016 hasta la –6005. Pero no nos preocupamos si no logramos entender todo, de cualquier manera podremos hacer andar nuestro programa. Las siguientes líneas son:

```
80 POKE -607,232
90 POKE -608,128                    CALL 59520
100 POKE -609,205
```

Estas tres líneas colocan una orden en un lugar por el que la máquina pasa muy seguido, exactamente cada 1/50 de segundo. La orden es la siguiente: realice el programa en Assembler. O sea que cincuenta veces por segundo la máquina leerá esa orden y correrá nuestro programita. Pero no tenemos de qué preocuparnos, pues la cantidad de tiempo que se pierde es mínima.

Luego de esto podremos realizar nuestro programa. Cuandoya estemos en un modo de pantalla que nos permita Sprites, hayamos definido los sprites y tengamos colocado uno en la pantalla con una orden PUT SPRITE podremos hacer lo siguiente:

```
300 POKE -6015, N * 4 + M
310 POKE -6009, K
```

En realidad no se debe teclear realmente M, N ni K, sino lo siguiente:

— En lugar de N, el número del sprite que deseamos mover.

— La M debe ser reemplazada por un 0 cuando se quiera darle el movimiento vertical (o directamente no poner nada en lugar de + M) y por un 1 cuando queremos darle movimiento horizontal.

— La K es la velocidad de desplazamiento. 1 es el movimiento más lento de los crecientes (hacia la derecha o hacia abajo) y 255 es el movimiento más lento decreciente (hacia la izquierda o hacia arriba). Los números intermedios dan velocidades más altas.

Lo que hacen estas dos líneas es cambiar al programa en Assembler, que hasta ahora no se refería a nada en especial, para que se refiera a un Sprite, una dirección y una velocidad concretas. Hasta ahora el programa ya estaba funcionando, pero sin modificar nada que podamos ver en la pantalla, o que importe en nuestro programa. Ahora los Sprites ya tienen movimiento.

Es muy importante anular esa orden que llama a nuestro programa Assembler cuando querramos volver a una pantalla de texto, si no queremos ver bellas obras de arte abstracto en la pantalla. Para eso basta con neutralizar a la línea 100 con la siguiente:

```
600 POKE -609,201
```

Finalmente, podemos darle a nuestros Sprites no sólo movimiento vertical u horizontal, sino también diagonal. Para eso modificaremos nuestro programa inicial, creando una segunda orden data similar a la de la línea 60. También podemos crear dos órdenes más como las de las líneas 300 y 310, porque hay que aclarar la dirección y la velocidad del movimiento vertical y el horizontal, que en realidad actúan en forma independiente, pero al hacerlo al mismo tiempo nos dan el efecto de un movimiento en diagonal. Aquí va un programa de ejemplo:

```
10 CLEAR 200, -6016
20 FOR A = -6016 TO -5994
30 READ A$
40 POKE A, VAL ("&H" + A$)
50 NEXT
60 DATA 21,0,1B,CD,4A,0,C6,0,CD,4D,0
70 DATA 21,0,1B,CD,4A,0,C6,0,CD,4D,0
80 DATA C9
90 POKE -607,232
100 POKE -608,128
110 POKE -609,205
120 SCREEN 2,1
130 SPRITE$ (1) = STRING$ (8,255)
140 PUT SPRITE 1, (127,100)
150 POKE -6015, 1 * 4+1
160 POKE -6009,2
170 POKE -6004,1 * 4
180 POKE -5998,3
190 A$ = INPUT$(1)
200 POKE -609,201
```

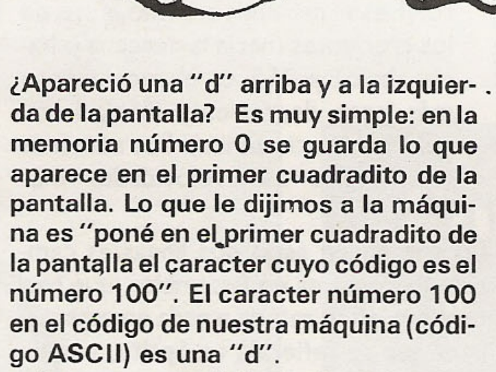
Entre las líneas 10 y 110 cargamos, protegemos y ponemos a andar al programa Assembler. En la 120 vamos a la pantalla de SPRITES, en la 130 creamos uno (un cuadrado) y en la 140 lo ponemos. En la 150 orientamos la primera parte del programa Assembler hacia nuestro Sprite en movimiento horizontal, y en la 170 cargamos la velocidad de ese movimiento (2). En la 170 y la 180 hacemos lo mismo para la segunda parte del programa y el movimiento vertical, con velocidad 3. La 190 sólo espera que toquemos una tecla: fijémonos que aunque no ejecutemos ninguna orden nuestro Sprite se sigue moviendo. En la 200 "apagamos" nuestro programa Assembler para volver a la pantalla de texto sin problemas.

En síntesis, esta es una forma rápida y sencilla de darle un movimiento oblicuo y automático a nuestro sprite, lo que les dará más velocidad y precisión a nuestros programas.

Gustavo FAIGENBAUN

¿QUE ES LA VRAM?

Para aprovechar al máximo las posibilidades del Basic, es muy importante no programar siempre con las mismas instrucciones.



Para descubrir las posibilidades ocultas de nuestra MSX que nos permitan hacer más cosas y más rápido hay que aprender a usar algunas instrucciones que no nos resultan muy familiares.

Entre esas instrucciones están VPEEK y VPOKE (Video PEEK y Video POKE).

Tecleemos lo siguiente:

```
VPOKE 1000,4
-RETURN-
```

Aparentemente no pasó nada. Ahora tecleemos:

```
PRINT VPEEK (1000)
-RETURN-
```

Si lo hicimos bien, en la pantalla apareció un 4. ¿Qué pasó? La orden VPOKE 1000,4 es la forma correcta de decirle a nuestra MSX "guardá el número 4 en la posición de memoria de video número 1000". La posición de memoria de video número 1000 es una pequeña variable, una variable que sólo puede guardar números entre 0 y 255. Si hubiéramos tecleado VPOKE 1000,300 se hubiera producido un error. PRINT VPEEK (1000) le indica a la máquina que escriba en la pantalla el contenido de la posición número 1000. Nuestra máquina tiene 16384 posiciones de memoria igualitas a la 1000 en donde guarda números entre 0 y 255. Todas estas posiciones juntas se llaman VRAM (Video Random Acces Memory). Cada una de estas "variables" tiene por nombre a un número entre el 0 y el 16383 y en todas ellas podemos escribir o leer datos con VPEEKS y VPOKES. Pero a diferencia de las verdaderas variables, los datos que guardamos en la VRAM tienen una función muy importante: en esos numeritos está escrito lo que vemos en la pantalla.

Por ejemplo:

```
SCREEN 0
-RETURN-
VPOKE 0,100
-RETURN-
```

¿Apareció una "d" arriba y a la izquierda de la pantalla? Es muy simple: en la memoria número 0 se guarda lo que aparece en el primer cuadradito de la pantalla. Lo que le dijimos a la máquina es "poné en el primer cuadradito de la pantalla el caracter cuyo código es el número 100". El caracter número 100 en el código de nuestra máquina (código ASCII) es una "d".

entremos pues el siguiente programa, y hagámoslo correr.

```
10 FOR A=0 TO 960
20 VPOKE A,32
30 NEXT
```

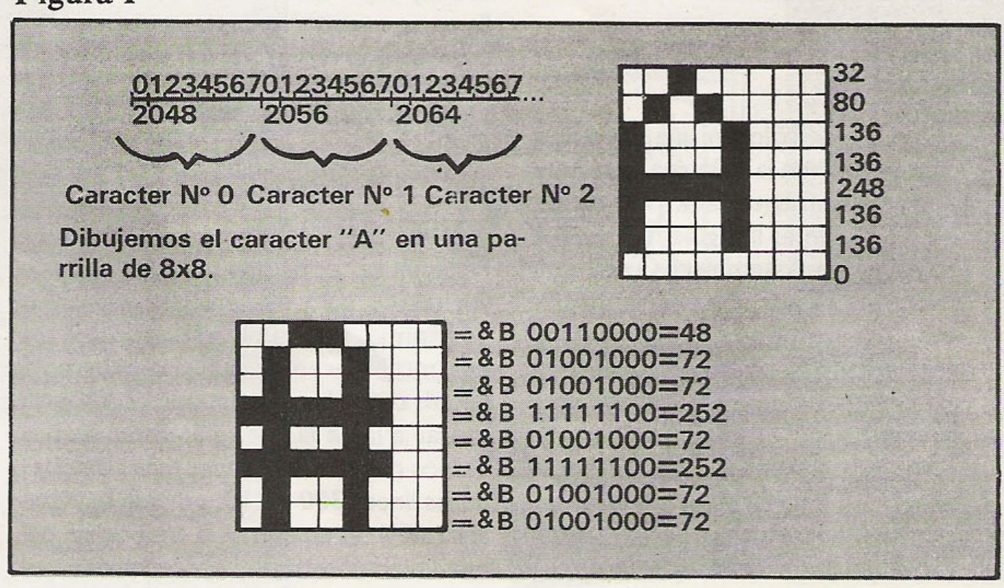
El caracter 32 es el espacio en blanco, y la "SCREEN 0" (pantalla número 0) tiene 960 posiciones. Por eso nos ha borrado todos los cuadraditos de la pantalla, incluso los de las teclas de función (último renglón) que no se van con un CLS normal. Para que aparezca en pantalla nuevamente el contenido de las teclas de función debemos pulsar SHIFT.

Veamos otro ejemplo:

```
10 FOR A=0 TO 960
20 B=VPEEK(A)
30 IF B    =97 THEN VPOKE A,B-32
40 NEXT
```

Una vez que hayamos entrado el programa, tecleemos unas cuantas letras

Figura 1



minúsculas y hagámoslo correr. Si todo va bien, todas las letras minúsculas de la pantalla serán cambiadas por mayúsculas, incluso las de las teclas de función. Lo que hace este programa es:

-En la línea 20, lee los datos de las posiciones de memoria y los guarda en la variable b.

- En la línea 30, se fija si el dato es mayor o igual que noventa y siete. Si lo es, en el cuadradito de la pantalla correspondiente hay una letra minúscula, porque los códigos de las minúsculas van desde 97 hasta 122 y los de las mayúsculas desde 65 hasta 90. Si la letra es minúscula la reemplaza por la mayúscula correspondiente, cuyo código es el de la minúscula menos 32.

```
  a=código 97      d=código 100
           −32               −32
  A=código 65      D=código 68
```

Borremos pues, el programa anterior y hagamos correr éste:

```
10 FOR A=0 TO 255
20 VPOKE A*2,A
30 NEXT
```

Los caracteres que aparecieron en la pantalla son todos los que podemos mostrar por este método, a menos que creemos caracteres nuevos. ¿Cómo hacerlo?

Observemos que para la imagen de la pantalla, hasta ahora sólo usamos 960 posiciones de memoria de las 16384 que dijimos que tenía la VRAM. El conjunto de estas 960 posiciones se llama "Tabla de imagen de pantalla". Pero además hay otra tabla que se llama "Tabla de diseño de caracteres", con la cual podemos redefinir, es decir, darle una forma distinta, a cualquiera de los caracteres de la máquina. La tabla de diseño de caracteres se encuentra desde la posición de memoria 2048. Ocho posiciones de memoria, es decir, ocho numeritos entre 0 y 255 nos dicen cómo está formado el caracter.

Igual que con los sprites, por cada línea tenemos un número entre 0 y 255 (que es el equivalente al número binario que se haría si cada cuadradito en blanco lo reemplazáramos por un 0 y cada cuadradito en negro por un uno).

Redefinamos la "A" para tener el signo del austral. (figura 1)

NOTA: Cuando estamos en "Screen 0" no podemos usar las dos últimas columnas de la parrilla.

Ya tenemos los datos. ¿En qué posiciones de memoria ponerlos? Para saber la posición inicial de la VRAM a partir de la cual entramos los ocho números tenemos que aplicar la siguiente fórmula:

2048+(código del caracter*8)

Por lo tanto, el listado del programa es éste:

```
10 A=2048+ASC("A")*8
20 FOR A=A TO A+7
30 READ B
40 VPOKE A,B
50 NEXT
60 DATA 48,72,72,252,72,252,72,72
```

Una vez que lo ejecutemos, vemos la nueva forma de las "A" de la pantalla. Finalmente, el siguiente programa nos permite ver "en gigante" cómo están hechos los caracteres.

```
10 FOR A=2048 TO 4096
20 B$=BIN$ (VPEEK(A))
30 B=LEN(B$)
40 B$=STRING$ (8−B,"0")+B$
50 FOR C=1 TO 8
60 IF MID$ (B$,C,1)="1" THEN PRIN
T

CHR$(219);ELSE PRINT"";
70 NEXT
80 PRINT
90 NEXT
```

Quedán aún muchas cosas para ver en Screen 0 y en otras pantallas. Gracias a los VPEEKs y VPOKEs podremos dibujar, diseñar y mover sprites, y muchas otras cosas.

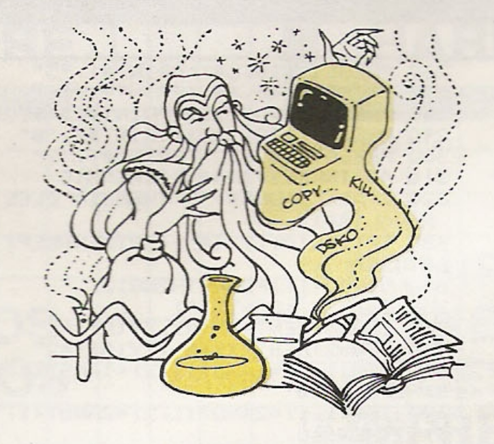
Gustavo Faigenbaum



CARACTERES INVERTIDOS

Con el programita de la figura 1, se invierten los caracteres. Tenemos la opción de invertir sólo algunos caracteres o bien todos. Al principio del programa, se preguntará qué caracteres queremos invertir, pero si escribimos la palabra "todos" (no interesa si está en mayúsculas o minúsculas), invertirá caracter a caracter todo el set. Esto tardará un poco (o mejor dicho bastante), pero vale la pena esperar.

El programa busca en memoria la definición del caracter a invertir. Lee los ocho bytes que lo definen y luego lee bit por bit de cada byte. Al realizar esta última lectura, cambia los unos por ceros. Es decir, si el byte a tratar es 10010111, entonces al invertirlo quedará: 01101000. Y el resultado de hacer el cambio de los ocho bytes de cada caracter producirá el efecto de invertirlos.



LIMPIEZA MISTERIOSA

Cada vez que hacemos un listado, las líneas del mismo suelen mezclarse con mensajes que ya existían en pantalla. Esto se puede salvar de dos formas distintas: definiendo alguna tecla de función para que limpie la pantalla y luego liste, o bien haciendo correr el programita de la figura 2.

Este utilitario primero limpiará la pantalla y luego listará las líneas del programa en memoria.

En realidad aquí se usan las dos rutinas Assembler de LIST y CLS. Copiemos el programa, hagámoslo correr y borremos línea por línea sin entrar NEW. Cada vez que querramos listar, se efectuarán las dos operaciones que mencionamos antes.

VELOCIDAD + AHORRO DE MEMORIA

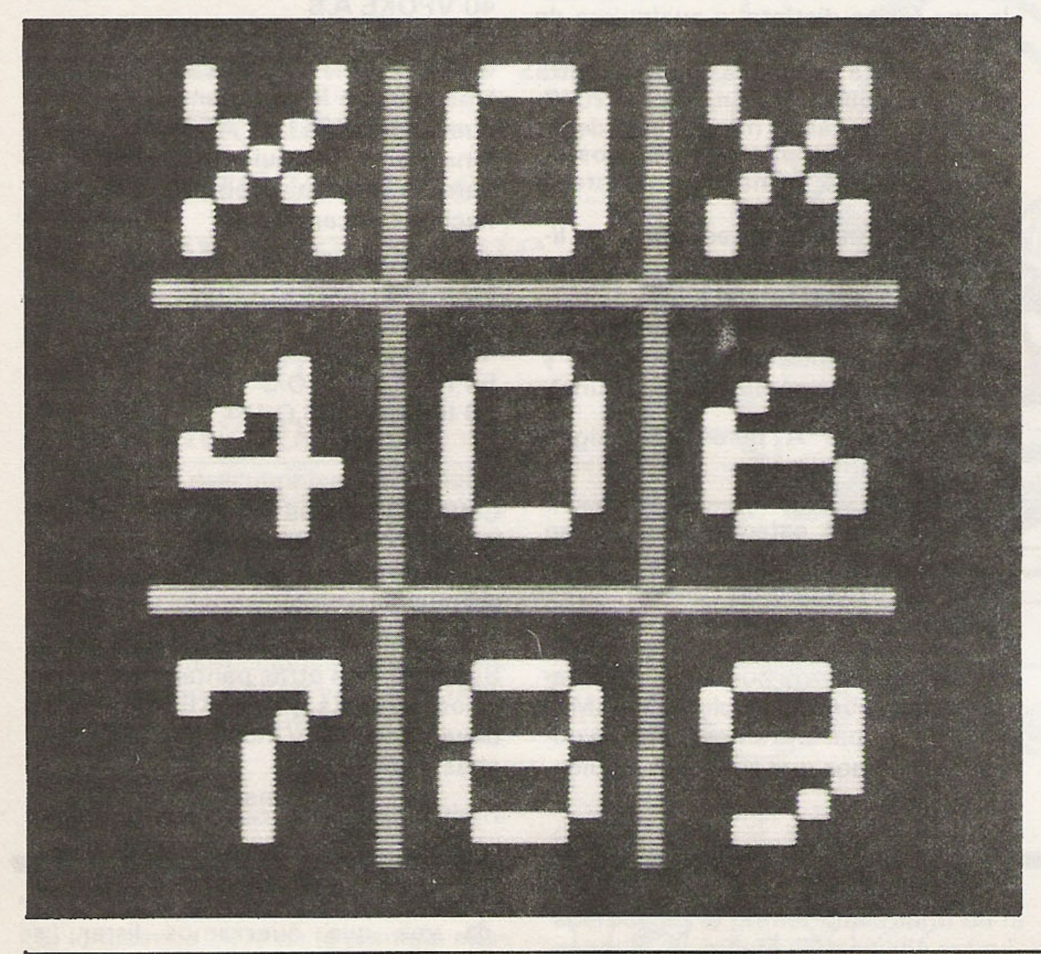
Para que nuestros programas se realizen con mayor rapidez, es conveniente tener en cuenta algunos secretos:

* eliminemos las instrucciones REM siempre y cuando ninguna instrucción GOTO salta a dicho REM. Y si reemplazamos esta instrucción por apóstrofe " ' ", ahorraremos memoria.
* comprimir las instrucciones (líneas con multisentencias) todo lo que sea posible. De esta forma el sistema ahorra tiempo porque no hace falta saltar a distintas líneas para ejecutar la siguiente instrucción.
* para pintar rectángulos, es más rápido hacerlo con LINE..., BF que con PAINT.
* tratar de evitar usar matrices cuando sea posible porque éstas consumen más tiempo que las variables.
* colocar las rutinas que llamamos varias veces al principio del programa principal. Así también se ahorra tiempo.



TATETI

Clase: juego



Este programa lo enviaron tres lectores de nuestra revista. Los autores fueron: Diego Ruiz, Axel Mori y su hermano Martín. Nuestros amigos han realizado un muy buen programa aplicando ventajas de las MSX como SCREEN 3, PLAY y LINE. Este entretenimiento es para dos jugadores.

Agradecemos a estos tres lectores por habernos enviado este listado porque demuestra que ellos también quieren compartir los programas que escriben.

VARIABLES IMPORTANTES

T(3,3)= coordenadas del tablero.
X(9)= jugadas del participante A.
Y(9)= jugadas del B.
XX(9)= coordenadas impresión de A.
YY(9)= coordenadas de impresión de B.
DD= jugadas generales.
W= ficha a jugar.
A$(1) y A$(2)= nombre de los jugadores.
A$= mensajes.

```
10 CLS:KEYOFF
20 '
                              --TA--TE
--TI--

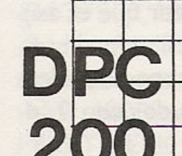
30 LOCATE15,0:PRINT"TA-TE-TI":LO
CATE10,1:PRINT"Hecho por 'AXDI'":
LOCATE10,3:PRINT"para K-64":LOCAT
E10,5:PRINT"COPYRIGHT '86"
40 FORI=1TO3000:NEXT
50 COLOR15,1,13:SCREEN3:OPEN"GRP
:"AS1
60 D=2:W=1
70 DIMX(9),Y(9),T(3,3)
80 A$="05BGB06CEG05BGB06CEG05BGB
"
90 B$="04BGB05CEG04BGB05CEG04BGB
"
100 C$="03BGB04CEG03BGB04CEG03BG
B"
110 FORI=1TO9:READ X(I),Y(I),XX(
I),YY(I):NEXTI
120 PLAY"T200L2XA$;","T200L2XC$;
"
130 PRESET(5,70):PRINT#1,"TA°TE°
TI"
140 FORI%=2TO15:LINE(0,40)-(256,
40),I%:LINE(0,150)-(256,150),I%:P
AINT(0,0),I%:PAINT(0,190),I%:NEXT
150 GOSUB 700
160 PLAY"T200L24XB$;"
170 SCREEN3
180 LINE(100,20)-(100,140),13
190 LINE(132,20)-(132,140),13
200 LINE(75,52)-(161,52),13
210 PRESET(78,20):PRINT#1,"1"
220 PRESET(110,20):PRINT#1,"2"
230 PRESET(140,20):PRINT#1,"3"
240 LINE(75,100)-(161,100),13
250 PRESET(78,64):PRINT#1,"4"
260 PRESET(110,64):PRINT#1,"5"
270 PRESET(140,64):PRINT#1,"6"
280 PRESET(140,112):PRINT#1,"9"
290 PRESET(110,112):PRINT#1,"8"
300 PRESET(78,112):PRINT#1,"7"
310 A$=INKEY$:IFA$=""THEN310
320 IFA$<"1"ORA$>"9"THEN310 ELSE
 A=VAL(A$)
330 IFT(XX(A),YY(A))<>0THENBEEP:
GOTO310
340 DD=DD+1:IFDD=9THEN810
350 S=XX(A):SS=YY(A)
360 IFW=1THENT(XX(A),YY(A))=1
370 IFW=2THENT(XX(A),YY(A))=2
380 IFT(1,1)=1ANDT(1,2)=1ANDT(1,
3)=1THEN660
390 IFT(1,1)=2ANDT(1,2)=2ANDT(1,
3)=2THEN660
400 IFT(1,1)=1ANDT(2,1)=1ANDT(3,
1)=1THEN660
410 IFT(1,1)=2ANDT(2,1)=2ANDT(3,
1)=2THEN660
420 IFT(1,1)=1ANDT(2,2)=1ANDT(3,
3)=1THEN660
430 IFT(1,1)=2ANDT(2,2)=2ANDT(3,
3)=2THEN660
440 IFT(1,2)=1ANDT(2,2)=1ANDT(3,
2)=1THEN660
450 IFT(1,2)=2ANDT(2,2)=2ANDT(3,
2)=2THEN660
460 IFT(1,3)=1ANDT(2,3)=1ANDT(3,
3)=1THEN660
470 IFT(1,3)=2ANDT(2,3)=2ANDT(3,
3)=2THEN660
480 LINE(X(A),Y(A))-(X(A)+20,Y(A
)+28),1,BF
490 IFT(1,3)=1ANDT(2,2)=1ANDT(3,
1)=1THEN660
500 IFT(1,3)=2ANDT(2,2)=2ANDT(3,
1)=2THEN660
510 PRESET(X(A),Y(A))
520 IFW=1THENPRINT#1,"X"
530 IFW=2THENPRINT#1,"O"
540 SWAPW,D
550 GOTO 310
560 GOTO 560
570 DATA 78,20,1,1
580 DATA 110,20,1,2
590 DATA 140,20,1,3
600 DATA 78,64,2,1
610 DATA 110,64,2,2
620 DATA 140,64,2,3
630 DATA 78,112,3,1
640 DATA 110,112,3,2
650 DATA 140,112,3,3
660 CLS:SCREEN0
670 PRINT "Te felicito ";A$(W)
680 PRINT "hiciste TA-TE-TI."
690 GOTO 840
700 SCREEN0
710 A$="Bienvenido al TA-TE-TI":
X=0:Y=0:GOSUB800
720 A$="Instrucciones...":X=0:Y=
1:GOSUB800
730 LOCATE0,3:INPUT"Nombre del j
ugador A:";A$(1)
740 LOCATE0,4:INPUT"Nombre del j
ugador B:";A$(2)
750 A$="Cada posición tiene un n
úmero":X=0:Y=6:GOSUB800
760 A$="púlselo para colocar su
ficha.":X=0:Y=7:GOSUB800
770 A$="-PRESIONE UNA TECLA-":X=
0:Y=15:GOSUB800
780 IFINKEY$=""THEN780
790 RETURN
800 FORP=1TOLEN(A$):LOCATEX+P,Y:
PRINTMID$(A$,P,1);:NEXTP:RETURN
810 CLS:SCREEN0
820 PRINT"Lo siento,empataron...
"
830 GOTO 840
840 PRINT:PRINT:INPUT"Juegan de
nuevo:";W$
850 IFW$="si"THENRUN
860 IFW$="no"THENEND
870 GOTO840
```

PORQUE LA COMPUTACION ES EL FUTURO...



MICROMATICA srl.

LOS PROFESIONALES DE LA COMPUTACION



DPC 200

ENTREGA INMEDIATA

SOFTWARE	HARDWARE
• JUEGOS	• EQUIPOS
• UTILITARIOS	• DISKETTERAS
• EDUCATIVOS	• ACCESORIOS
• A MEDIDA	• IMPRESORAS

BIBLIOGRAFIA - CURSOS (NIÑOS, ADULTOS, PROFESIONALES)

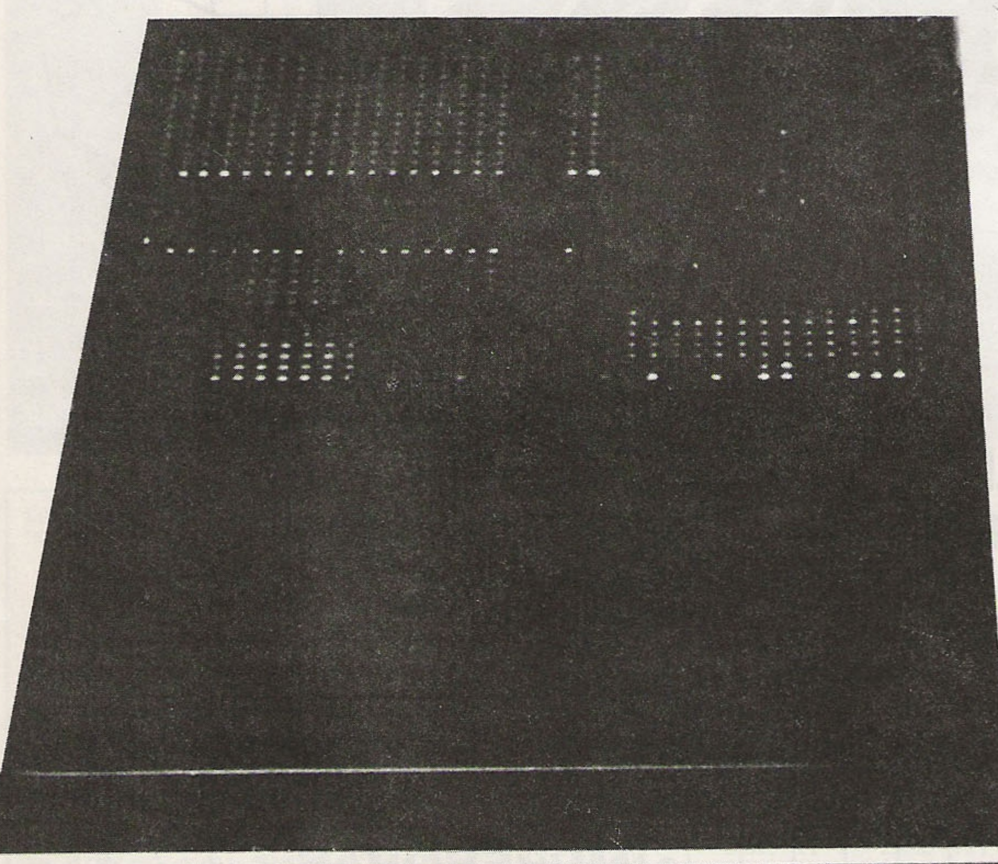
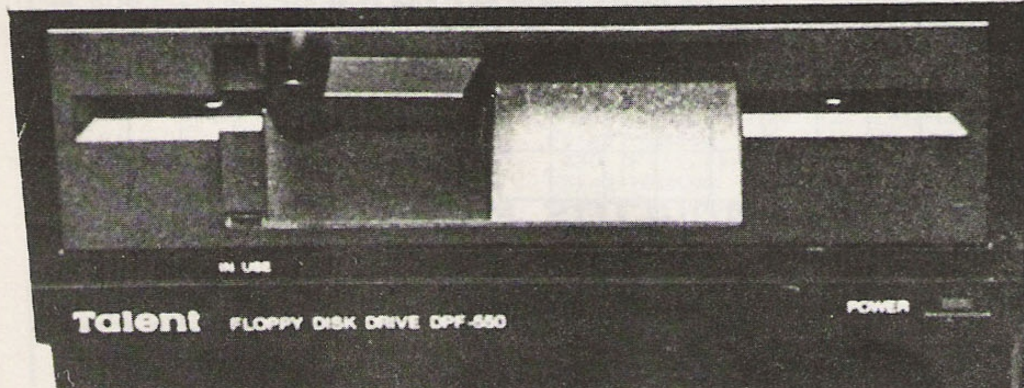
DISTRIBUIDORES OFICIALES

Talent MSX

AV. PUEYRREDON 1135 (1118) Tel.: 821-5578

EL SISTEMA OPERATIVO DOS

El dos no es cualquier cosa, sus comandos no son precisamente pocos. Comenzaremos viendo algunas funciones complementarias a los comandos disponibles de este sistema operativo, desde BASIC.

CVI, CVS, CVD: Estas tres instrucciones cumplen la función de convertir valores alfanuméricos en numéricos, que se encuentren archivados de forma tal que puedan ser leídos en forma aleatoria por las instrucciones asociadas a este modo.

CVI significa: convertir en valor entero. Por este motivo sólo como argumento soporta dos bytes cualesquiera, del archivo que se esté tratando.
CVS significa: convertir en valor de simple presición. Así es como soporta 4 bytes como argumento, que deben, sin lugar a dudas, ser leídos del archivo de acceso aleatorio que se está utilizando.
CVD significa: convertir a valor de doble presición. Entonces, por las características de este tipo de números, soporta como argumento a ocho bytes. En todos los casos estos bytes figuran bajo el nombre de alguna variable alfanumérica.
Veamos un ejemplo. Supongamos que dentro de un programa, en el que se abrió un archivo de acceso aleatorio (que por supuesto, figuraba en el disquette), figuran las siguientes líneas:

```
50 FIELD#1, 2 AS A$, 4 AS B$, 8 AS C$
60 GET#1
70 ENTERO = CVI (A$)
80 SIMPLE = CVS (B$)
90 DOBLE = CVD (C$)
```

Como sabemos (quiero creer que es así) la instrucción FIELD ha habilitado al buffer para aceptar tres campos llamados A$, B$ y C$ cuyas longitudes son 2, 4 y 8 respectivamente.
¡Oh casualidad!, corresponden a las longitudes de los strings o cadenas alfanuméricas aceptables por las tres funciones que vimos antes. Seguramente algo nos traemos debajo del poncho.

Y así es, nuestra sagaz mente ha preparado de esta forma al buffer para asignar a cada valor leído un tipo de variable distinto y de características numéricas. El resto del programa es claro y lo comprendemos perfectamente si es que hemos leído los números anteriores.

DSKF: esta función es de utilización simple, y sirve para averiguar la cantidad de bytes que quedan libres en un determinado disquette, en unidades de kilo.
El argumento que necesita para alimentarse es el número correspondiente al drive en que se encuentra dicho disquette.
Estos números identifican a los drives o unidades de disco de la siguiente forma:
0 corresponde al último drive con que se trabajó, sin importar cual de ellos era.
1 corresponde al identificador del drive A.
2 corresponde al identificador del drive B.
Por lo visto, un ejemplo válido sería: PRINT DSKF(2) que imprimirá en pantalla la cantidad de kbytes libres en el disquette del drive B.

DSKI$: como dijimos en el número 1 de esta publicación, al referirnos a la instrucción DSKO, un disco es formateado para soportar 360 kbytes de información repartidos en sectores de 512 bytes.

En cierta forma esta función recorre el camino inverso al de la función DSKO. Pues, dado un nombre de drive y un número de sector válido, ésta transfiere dicho sector al bloque de memoria al cual apuntan el par de bytes F351H y F352H.

Supongamos que queremos ver el contenido del sector 5 de un disco que contiene cierta información y se aloja en el drive A.

En primer lugar deberemos elegir en que lugar de la memoria queremos ubicar esos 512 bytes de información, asumamos que éste es el valor 40000 decimal. Entonces haremos que F351 y F352 "apunten" a esa dirección de la memoria por medio de los siguientes comandos: POKE &HOF351,64 y POKE &HOF352,156.

Así, si efectuamos la siguiente verificación obtendremos como respuesta el valor 40000:

```
PRINT PEEK (&HF351) + 256*PEEK
(&HF352).
```

Ahora sí, podemos efectuar la instrucción DSKI$ (1,5) y el bloque correspondiente al sector 5 del drive 1 (A) habrá pasado a la memoria a partir de la posición 40000.

Para verificar este último proceso podríamos efectuar el siguiente programúsculo:

```
10 FOR F=40000 TO 40511:
PRINT PEEK (F);: NEXT
```

Para ver algo más interesante aún, podríamos cambiar la instrucción PEEK del programa anterior por CHR$ (PEEK(F)); y así veríamos la representación ASCII de los valores que se encontraban en ese sector.

Obviamente tendremos algunos problemas cuando este comando interprete a los valores que corresponden a caracteres especiales como Line Feed, Return o Home. Por este motivo será mucho más útil separar el programa en diferentes líneas con el fin de insertar una que contenga una instrucción IF que haga traducir solamente a los caracteres que corresponden a símbolos legibles.

Así veremos que en esa pantalla sucia aparecen algunos nombres que sólo nosotros reconoceremos.

Este truquito fue de regalo, todo por el mismo precio.

EOF: significa fin de un archivo (End Of File). Y así es nomás, nos indicará si un archivo de texto secuencial se ha terminado.

Por supuesto que para indicar el fin del archivo deberemos estar leyéndolo.

Así devolverá el valor −1 si es que esto ocurre.

Veamos un ejemplo:

```
10 OPEN "ARIEL.PAT" FOR INPUT AS#1
20 IF EOF(1) THEN GOTO 1000
30 :
:  :
:  :
```

La línea 20 pasará el control a la línea 1000 cuando el archivo se haya terminado. Por supuesto, debemos pasar el control a la línea donde se encuentra esta instrucción, antes de hacer cada lectura de un registro.

El número que figura entre paréntesis luego de EOF, es el que se corresponde con el número de archivo que se ha abierto y se quiere comprobar.

INPUT$: se lo utiliza tanto para leer un dato determinado desde el teclado como desde un archivo secuencial.

Su sorprendente característica (*'§:(?) es la de leer todos los caracteres que se le presenten, salvo el Control-STOP. Para ver las diferencias del INPUT sólo, veamos la primer entrega de esta publicación. Su utilización y sintáxis es idéntica a ésta, se escribe INPUT$#n, donde n es el número de archivo secuencial que se ha abierto para lectura de datos.

Sin el símbolo de numeral, asumirá que nos estamos refiriendo a la entrada de datos desde el teclado. Por supuesto, siempre debe ir acompañado de una variable de algún tipo conocido.

LOC: esta función se usa tanto en el tratamiento de archivos abiertos para tratamiento aleatorio como para secuencial.

Nos indica el número de registro que está listo para ser leído o escrito por medio de las instrucciones destinadas a tal fin.

Debe seguirse del número de archivo encerrado por paréntesis, al que se hace referencia.

LOF: retorna el tamaño en bytes, del archivo al que hace referencia por medio del número encerrado entre paréntesis, que lo sucede.

MKI$, MKS$ y MKD$: se las utiliza para convertir valores numéricos enteros, de simple presición y de doble precisión respectivamente a valores alfanuméricos.

Se las suele usar en conjunción con las instrucciones LSET y RSET. Algo así: LSET D$=MKS$ (valor o variable numérica) donde D$ corresponde al nombre de un campo definido en el buffer por medio de FIELD (por ejemplo) para un fin determinado.

Estas son las últimas funciones que veremos y que pueden accederse desde Basic.

Lo que sigue es una lista descriptiva de una serie de comandos disponibles desde el mismísimo DOS.

Recordemos que la instrucción CALL SYSTEM era la llave para ingresar al mundo de fantasía de los archivos.

En este celeste e infinito espacio de comandos increíbles, podremos comandar esta poderosa nave visitando, observando, sacando información y modificando cada uno de los archivos que pueblan esta informática galaxia.

Una vez que hayamos entrado en este universo, ya desde BASIC o por haber inicializado el sistema, nuestro indicador de drive activo estará permanentemente cumpliendo su función.

Así veremos todo el tiempo alguno de los siguientes indicadores:

A > o B >.

Entonces si nos proponemos trabajar con el disco que se encuentra en el drive B, y nuestro indicador es 'A>' deberemos efectuar la siguiente operación: A>B:, donde A> es el indicador, así que solamente deberemos digitar B: y Return.

Comencemos viendo algunos comandos.

BASIC: entrando solamente esta palabrita, luego de la 'A>' pasaremos al conocido universo del intérprete Basic en el cual ya estamos acostumbrados a viajar.

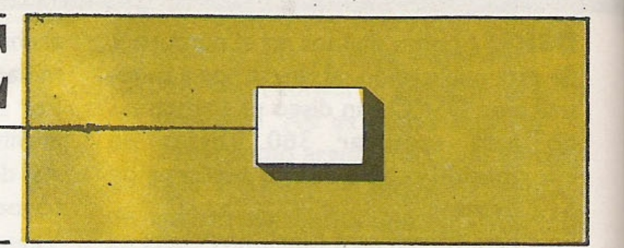
COPY: esta es la instrucción más discutida de este sistema, pues se usa para lo que ya estamos intuyendo: copiar uno, alguno o simplemente todos los archivos, incluyendo los programas, tanto en Basic como en Assembler, que se encuentren en un disco en otro soporte magnético de las mismas características, entre otras cosas.

En realidad se lo utiliza para copiar algún archivo de cualquiera de los periféricos, incluyendo el teclado, a otro periférico, incluyendo la impresora y el monitor.

Para comenzar a estimular nuestro enano pirata probemos la siguiente forma de esta potente instrucción:

COPY *.* B: Una vez que hayamos descubierto su uso y aplicación y estemos dispuestos a conocer las otras formas y posibilidades de esta instrucción, y del DOS, consigámonos un confortable almohadón para sentarnos frente al kiosco de revistas más cercano, porque como dice un osado y prestigioso periodista, por ahora lo dejamos ahí.

CONCURSO DE PROGRAMAS



auspiciado por TELEMATICA S.A. que proveerá los siguientes Premios:

PRIMER PREMIO

Un periférico (a elección entre un monitor, una diskettera y una impresora). **Una Beca** para trabajar en el Departamento de Investigación y Desarrollo de Telemática S.A.

SEGUNDO PREMIO

Un periférico (a elección entre un monitor, una diskettera y una impresora).

En caso de que el ganador no pueda utilizar la beca será ofrecida a quien obtenga el segundo premio, y si este tampoco pudiera aprovecharla se otorgará a alguno de los participantes del certamen que se hubiera destacado.

ESPECIAL Entre los programas recibidos, algunos de ellos podrán ser editados por Prosoft, reconociéndose los derechos de autor.

JUEGOS:

Temas:

- **TRUCO.** Premiaremos al programador que logre la mejor versión de este tradicional juego de salón.
- **DE INTELIGENCIA.** Los juegos que nos inspiren podrán ser "El Estanciero" o temas originales que sigan la línea

EDUCATIVOS:

Tema: • **LIBRE.**

UTILITARIO:

Tema: • **INTELIGENCIA ARTIFICIAL.** Se trata de lograr un programa que realmente nos haga discutir sobre el límite entre la simulación y la inteligencia de la computadora.
Obviamente la única forma de lograr esto será siguiendo las reglas propuestas de la Inteligencia artificial. Por esto se considerarán para premiar esta categoría, además de las reglas detalladas más abajo, la capacidad de auto-aprendizaje del programa, el nivel de inferencia del mismo, la capacidad y modo de almacenamiento de su base de datos, y principalmente su analizador sintáctico dado que hasta el momento no se ha logrado uno que dé suficiente credibilidad de que estamos frente a un ser racional que entiende nuestro idioma y no frente a una máquina a la que debemos hablarle con verbos en infinitivo al mejor estilo Tarzán (sin menospreciar a este último).

PROFESIONAL O GESTION:

Tema: • **LIBRE.** Dentro de este tema podrán figurar programas de las más diversas aplicaciones.

BASES: No sólo será indispensable que el programa enviado funcione correctamente sino que además debe cumplir con ciertas reglas.

- *Programación estructurada en bloques fácilmente diferenciables.*
- *Fácil seguimiento del mismo y detalle de éste como parte de su documentación. (Diagrama de bloques con los números de línea que los identifiquen).*
- *Aclaración y clara explicación de los algoritmos utilizados, deben figurar como parte de la documentación.*
- *Las variables y/o direcciones de memoria utilizados también se deben incluír en esta documentación.*
- *Listado de nemónicos assembler y la localización en memoria si es que se utiliza este tipo de lenguaje.*
- *Calidad y originalidad de gráficos, sonidos y pantallas de menú.*
- *El programa debe ajustarse a alguno de los temas propuestos más arriba. Esto es ELIMINATORIO.*

Los trabajos deberán enviarse antes del 30 de noviembre próximo (cierre del certámen) a: Paraná 720, piso 5º, (1017) Capital Federal.



MOVIMIENTO ARMONICO

Clase: Utilitario

Por el título, los que tienen el mismo vicio que Einstein, se habrán dado cuenta de qué se trata. Por otra parte el programa ofrece una clara explicación del tema.

Nos queda por decir que es una lograda aplicación del movimiento armónico producido por masas en suspensión. De no conocer el tema, por la explicación que el mismo programa da, quedaremos atrapados por la interesante demostración que ofrece.

También será de utilidad palos alumnos de escuelas técnicas especializadas en electrónica, y que estén viendo el desarrollo de las figuras del benemérito señor Lissajous.



Estructura del programa

100-285 Presentación y explicación

290-440 Estructura principal

460-580 Subrutina de movimiento

```
100 REM **MOVIMIENTO ARMONICO**
105 REM ├├├├├ LOAD MSX ┬┬┬┬┬┬
110 SCREEN 0
115 COLOR 1,14
120 KEY OFF
125 LOCATE 10,0
130 PRINT "MOVIMIENTO ARMONICO":FOR F=1 TO 1000:NEXT
135 LOCATE 10,0
140 PRINT STRING$(17,223):
145 PRINT" MIRAR UN LIBRO DE FIGURAS LISSAJOUS,PUEDE TRADUCIRCE A OBSERVAR DIBUJ
OS  RAROS PERO ATRAYENTES."
147 PRINT
150 PRINT" ESTOS DIBUJOS SON SIMPLES MOVIMIEN- TOS ARMONICOS INDEPENDIENTES, QUE
 SE MUEVEN ATRAVEZ DE DOS ABSISAS."
155 PRINT
160 PRINT" ESTO PUEDE PARECER COMPLICADO, PERO LA RAZON POR LA QUE SE PRODUCEN E
S   SIMPLE,PERO SUJETA A INFINITAS VARIA-CIONES DE SUS PARAMETROS."
165 GOSUB 180
170 PRINT " IMAGINESE DOS PENDULOS: UNO MOVIEN- DOSE EN DIRECCION NORTE-SUR, EL
OTRO EN DIRECCION ESTE-OESTE."
173 PRINT
175 PRINT" COMO ELLOS OSCILAN,EL PUNTO MAS      DISTANTE DEL PENDULO N.S Y EL PUN
TO  DEL QUE SE MUEVE EN DIRECCION E.O,    PUEDEN UNIRSE FORMANDO UN DIBUJO: ├├├├├
 FIGURA DE LISSAJOUS.├├├├├":GOSUB 180:GOTO215
180 LOCATE 9,19
185 PRINT STRING$(19,223)
190 LOCATE 7,20
195 PRINT "PRESIONE LA TACLA DE ESPACIO                     PARA CONTINUAR"
200 IF INKEY$<>"" THEN 200
205 IF INKEY$<>" " THEN 205
210 CLS:RETURN
215 PRINT "EL DIBUJO PRODUCIDO DEPENDE DE:"
220 PRINT "1. - EL PERIODO DE RAZON O PROPORCION"
223 PRINT
225 PRINT  " LA RELACION DEL TIEMPO QUE TARDA     CADA PENDULO EN COMPLETAR UNA
    OSCILACION"
230 PRINT:PRINT:PRINT:PRINT
235 PRINT "2. - EL ESCALON DE DISCREPANCIA":PRINT:PRINT
240 PRINT" POCICION RELATIVA DE LAS LENTEJAS DELOS PENDULOS CUANDO LA OSCILACION
    COMIENZA"
245 GOSUB 180
250 PRINT " EL ESCALON DE DISCREPANCIA GENERAL- MENTE ESTA TOMADA EN GRADOS, POR
 LO  TANTO UN ESCALON DE DISCREPANCIA DE  90 SIGNIFICA QUE AL COMIENZO"
255 PRINT"UN PENDULO ESTABA 1/4 DE OSCILACION  MAS ARRIBA DEL OTRO, MIENTRAS 180
    GRADOS SIGNIFICA QUE UN PENDULO       ESTABA 1/2 DE OSCILACION ARRIBA."
260 PRINT:PRINT
265 PRINT " ESTA SIMULACION COMPUTARIZADA        MOSTRARA GRAFICAMENTE EL CAMINO
    PERIODICO DE RAZON Y ESCALON DE       DISCREPANCIA COMBINADAS PARA DETER-  M
INAR EL TRAZADO DE LA FIGURA DE      LISSAJOUS."
270 LOCATE 2,20
275 PRINT" PRESIONE LA BARRA DE ESPACIO PARA    CORRER LA SIMULACION"
280 IF INKEY$<>"" THEN 280
285 IF INKEY$<>" " THEN 285
290 COLOR 1,14,4
295 SCREEN 2
300 OPEN "GRP:" FOR OUTPUT AS#1
305 '
310 SPRITE$(1)=STRING$(3,16)+CHR$(238)+STRING$(3,16)
315 SPRITE$(2)=STRING$(7,128)
320 SPRITE$(3)=STRING$(2,252)
325 '
```

```
330 PI=3.14159265#
335 S=PI/30
340 '
345 C(1)=12:C(2)=6:C(3)=13:C(4)=1
350 '
355 FOR W=0 TO 16
360 '
365 '
370 READ P,Q,I
375 GOSUB 445
380 NEXT W
385 '
390 '
395 T=RND(-TIME)
400 RESTORE 650
405 W=RND(1)*4
410 FOT T=0 TO W
415 READ Q
420 NEXT T
425 P= INT(RND(1)*(Q-1))+1
430 I=(INT(RND(1)*13))/2
435 GOSUB 445
440 GOTO 400
445 '
450 '
455 '
460 IF Q<0 THEN Q=-Q:FOR T=0 TO 2000:NEXT:CLS
465 F=P/Q
470 L=PI*Q*2
475 C=C(U):U=(U+1)MOD 5
480 R=PI*I/6
485 '
490 FOR T=121 TO 159
495 VPOKE T,0
500 VPOKE T+104,0
505 NEXT T
510 PSET (18,1),14
515 PRINT #1,"RAZON=";RIGHT$(STR$(P),2);":";RIGHT$(STR$(Q),(LEN(STR$(Q))-1))
520 PSET (125,1)
525 PRINT#1,"DISCREPANCIA=";30*I
530 '
535 FOR A=0 TO L STEP S
540 X=128+110*COS(A)
545 Y=99+79*COS((A*F)+R)
550 IF A THEN LINE-(X,Y),C
555 PUT SPRITE0,(X-3,Y-3),1,1
560 PUT SPRITE 1,(X,183),1,2
565 PUT SPRITE 2,(246,Y),1,3
570 PSET (X,Y),C
575 NEXT A
580 RETURN
585 '
590 '
595 '
600 DATA 1,1,0,1,1,1
605 DATA 1,1,3,1,1,5,1,1,6
610 DATA 1,-2,0,1,2,1
615 DATA 1,2,1,5,1,2,2,1,2,3
620 DATA 1,-1,0,1,3,0,1,5,0
625 DATA 1,-5,0,2,5,0
630 DATA 3,5,0,4,5,0
635 '
640 '
645 '
650 DATA -2,-3,-5,-7
```

MSX-LOGO

Se trata del programa que nos permitirá trabajar en LOGO, lo acompaña un manual por demás explicativo, del cual ofrecemos un anticipo, escrito por Hilario Fernández Long y Horacio Reggini.



Esta obra introduce al lector a una nueva implementación de Logo.

MSX-Logo significa otro hito en el camino del desarrollo de versiones de Logo cada vez más poderosas. Incluye la posibilidad de dibujar en la totalidad de la pantalla, pintar o sombrear recintos, utilizar distintos colores simultáneamente, el control de tortugas móviles, la manipulación de palabras y listas, el empleo de listas de propiedades y la generación de música.

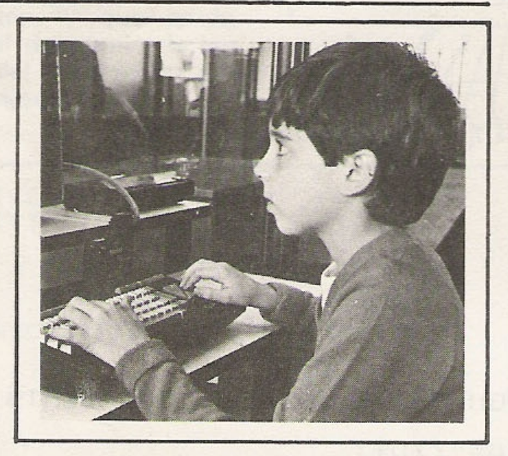
Pueden activarse treinta tortugas simultáneamente (antes llamadas "actores" en otras versiones de Logo), con distintas figuras y colores y cada una de ellas puede dejar rastro al moverse. Existen además las órdenes cuando y al contacto, que desencadenan una serie de acciones si se produce un suceso determinado. Es como si un diablillo o "demonio" estuviera al acecho mientras se ejecuta un procedimiento cualquiera; cuando ve que se produce el hecho especificado, hace que se cumplan las órdenes consignadas.

La sigla MSX designa un conjunto de normas que hacen posible que un programa pueda operar en distintos modelos de computadoras, siempre que hayan adoptado las normas MSX citadas.

POR QUE LOGO

El aprendizaje del lenguaje Logo es una de las puertas más interesantes para acceder al mundo de la computación. Su sencillez, su potencia y la posibilidad de emplearlo en todos los campos de la actividad humana pueden llegar a convertirlo en el lenguaje de programación más difundido y aceptado en los próximos años.

La potencia de logo se debe a tres razones fundamentales. En primer lugar, sus principios son fáciles de aprender y



los que se inician logran de inmediato resultados interesantes, sin necesidad de preparación esmerada. Por ello, incluso los niños pueden usarlo y, de hecho, millares de ellos lo están haciendo actualmente. Es por esta circunstancia que las computadoras, con Logo, se convierten en herramientas útiles para el proceso educativo. Sin embargo, Logo no es un lenguaje solo para niños. Esta imagen, que existe en algunos círculos, se debe al hecho de que los primeros ejemplos de su empleo ilustran procedimientos elementales para niños y/o principiantes adultos carentes de conocimientos previos de computación. Utilizarlo luego en aplicaciones importantes requiere, por supuesto, estudios organizados.

La segunda razón reside en su "extensibilidad", que hace de él un lenguaje descriptivo de extraordinaria utilidad. Así mismo, la formulación de programas en Logo no solo imparte una jerarquización lógica, sino que hace sencilla la búsqueda de los errores que inexorablemente aparecen en la programación.

La tercera razón tiene que ver con el uso de estructuras de datos muy generales como las listas, y las posibilidades que brinda la recursión.

Las nuevas versiones de Logo disponibles, como la presente, nos acercan hoy más a la posibilidad de su uso en campos diversos.

LAS COMPUTADORAS EN LA EDUCACION

El uso de Logo en la educación presupone una visión distinta de las máquinas. Logo se relaciona íntimamente —como modalidad educativa— con la creatividad y la libertad, valores que hay que reecontrar y reconstruir permanentemente en todo ámbito.

La implementación de Logo en las aulas no puede improvisarse, sino que requiere por parte de los docentes y de los directivos un reconocimiento cabal de los principios congnitivos en que está inspirado. Los educadores deben comprender que la filosofía de su uso es más importante que los productos que se pueden obtener de la máquina. Como instrumento delicado, Logo torna más rico y eficaz el proceso educativo pero, al mismo tiempo, lo vuelve más complicado y difícil de encaminar si no existe una concepción clara de sus fundamentos y alcances.

Es preciso comprender que un aprendizaje verdadero y profundo solo se alcanza cuando el conocimiento formal se relaciona con el conocimiento personal de manera variada y diversa. En un proceso educativo genuino, los temas tratados deben poseer puntos de contacto con el mundo circundante y relevancia personal para el alumno. El diálogo enseñanza-aprendizaje debe alentar y favorecer la actividad autoconstructiva del educando, a fin de que éste se encuentre con su propio yo, saque partido de sus recursos intelectuales y perciba que lo que aprende lo habilita para interactuar con su alrededor. Deben existir lazos de afectividad con lo que se aprende; es necesario que lo conceptual se vincule claramente con lo instrumental y que todo ambiente de aprendizaje permita convertir las ideas en hechos, con la posibilidad de corregir errores y de modificar rumbos. Logo solo puede arraigar y crecer en un ámbito donde se aceptan y practican las premisas anteriores.

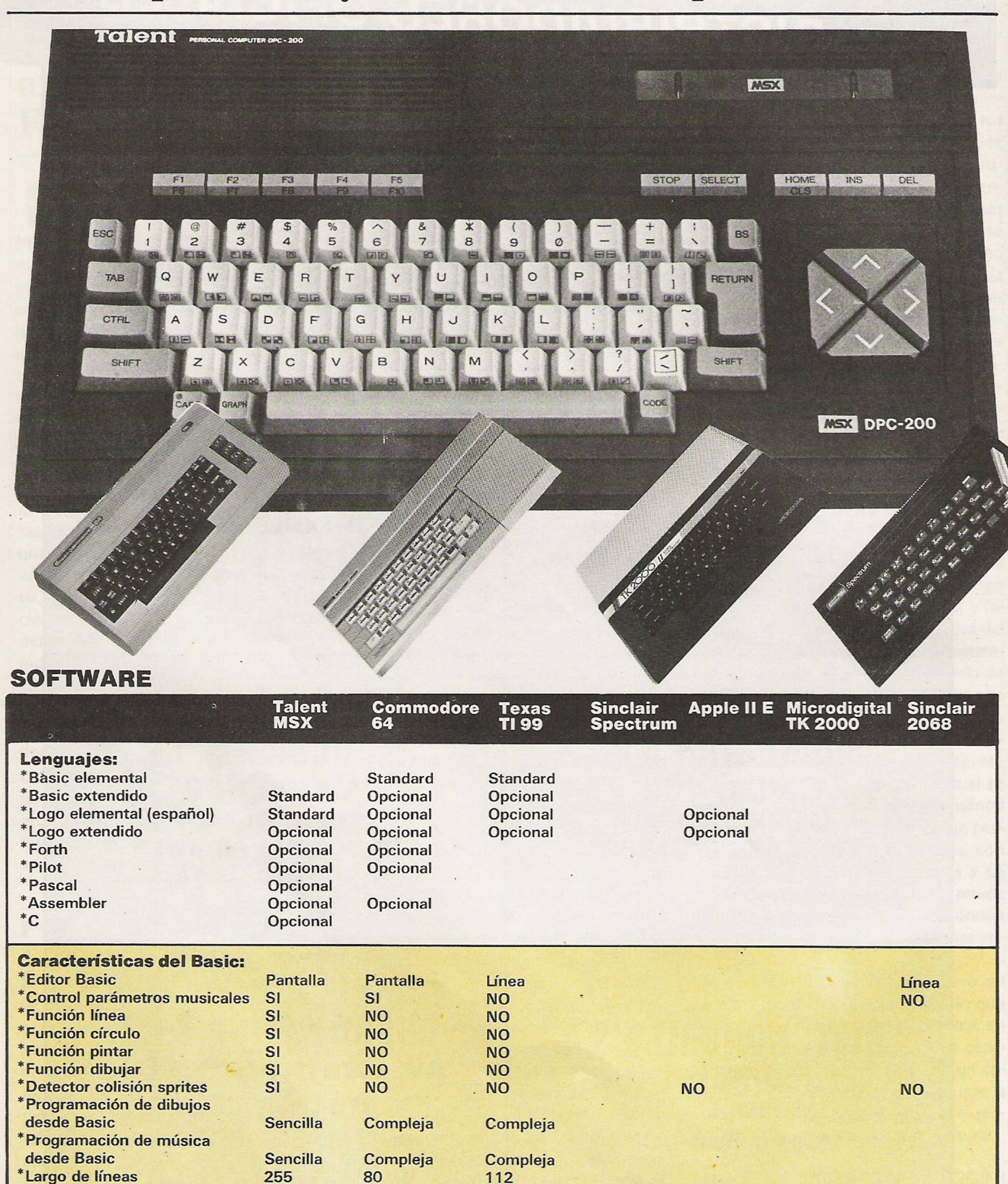
Logo puede ser considerado como un aporte filosófico a la educación* lo que agrega a sus posibilidades poderosas en el campo de la computación, una componente social trascendente. En un camino del pensamiento que busca que la ciencia y la tecnología estén más al servicio de las personas y de la cultura.

HORACIO C. REGGINI

* Desafío a la mente, Seymour Papert, Buenos Aires, Galápago, 1981, Dist. Emecé.

MSX DPC-200 VS. EL RESTO

El siguiente es un cuadro comparativo de las especiales características de la primera máquina lanzada por Talent y el resto de las computadoras.



SOFTWARE

	Talent MSX	Commodore 64	Texas TI 99	Sinclair Spectrum	Apple II E	Microdigital TK 2000	Sinclair 2068
Lenguajes:							
*Basic elemental		Standard	Standard				
*Basic extendido	Standard	Opcional	Opcional				
*Logo elemental (español)	Standard	Opcional	Opcional		Opcional		
*Logo extendido	Opcional	Opcional	Opcional		Opcional		
*Forth	Opcional	Opcional					
*Pilot	Opcional	Opcional					
*Pascal	Opcional						
*Assembler	Opcional	Opcional					
*C	Opcional						
Características del Basic:							
*Editor Basic	Pantalla	Pantalla	Línea				Línea
*Control parámetros musicales	SI	SI	NO				NO
*Función línea	SI	NO	NO				
*Función círculo	SI	NO	NO				
*Función pintar	SI	NO	NO				
*Función dibujar	SI	NO	NO				
*Detector colisión sprites	SI	NO	NO		NO		NO
*Programación de dibujos desde Basic	Sencilla	Compleja	Compleja				
*Programación de música desde Basic	Sencilla	Compleja	Compleja				
*Largo de líneas	255	80	112				
*Compatibilidad CP/M	SI	Opcional	NO				

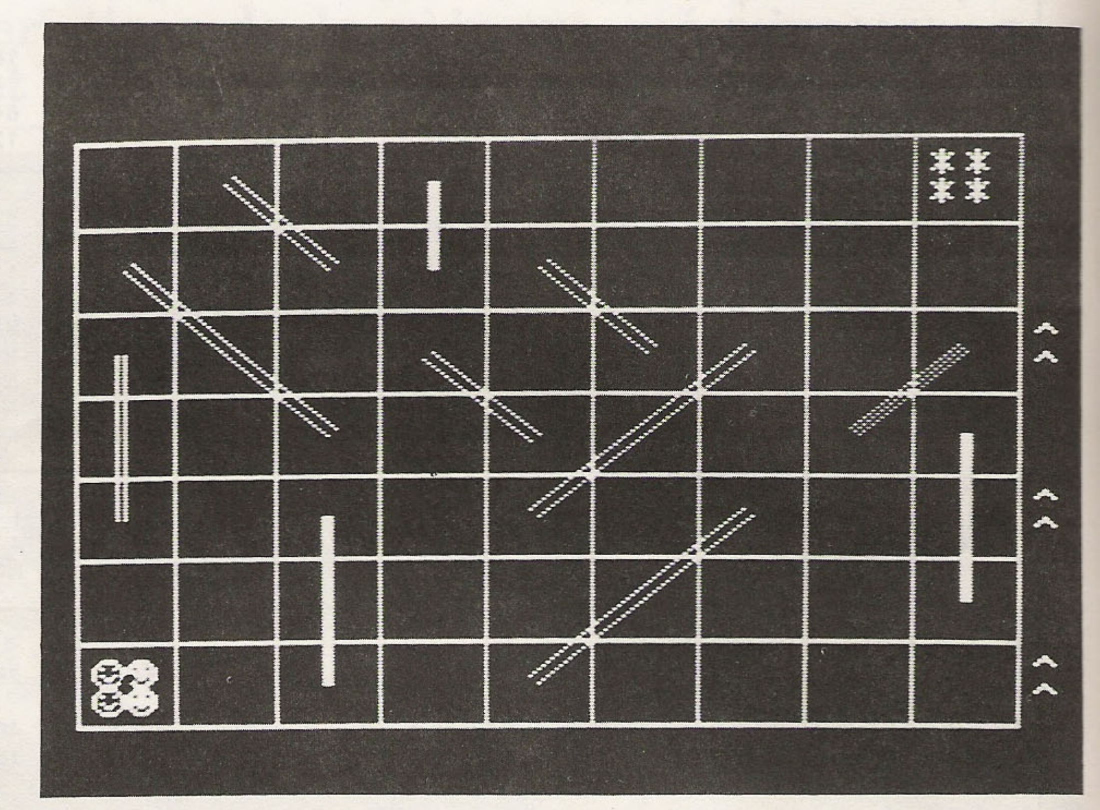
HARDWARE

	Talent MSX	Commodore 64	Texas TI 99	Sinclair Spectrum	Apple II E	Microdigital TK-2000	Sinclair 2068
Procesador	Z-80 A	6510	TMS 9900	Z-80	6502	6502	Z-80
Reloj	3,6 MHz	1 MHz		3,5 MHz	1 MHz	1 MHz	3,6 M Hz
Capacidad de memoria:							
*ROM incorporada	32 KB	20 KB		8 KB	16 KB	16 KB	24 KB
*ROM incorporada video	16 KB						
*RAM incorporada	64 KB	64 KB	16 KB	48 KB	64 KB	64 KB	48 KB
*RAM ampliable hasta	1 MB		48 KB		128 KB	128 KB	1,3 MB
Características teclado:							
*Cantidad total teclas	73	66	48	40	62	54	40
*Cantidad teclas definibles	10	8	—	—	—	—	—
*Cantidad teclas cursor	4	4	4	4	1	4	4
*Cantidad direcciones cursor	8	4	4	4	4	4	4
*Teclas p/may./min.	SI	SI	SI	SI	SI	NO	SI
*Teclas p/gráficos	SI	SI	NO	SI	NO	SI	SI
*Teclas p/editor textos	SI	NO	NO	NO	NO	NO	NO
*Buffer teclado	39 c	10 c	NO	NO	NO	NO	NO
*Tipo teclas	Mecánico	Mecánico	Mecánico	Goma	Mecánico	Mecánico	Mecánico
Características sonido:							
*Canales sonido	3	3	3		1		4
*Octavas por canal	8		8		4		8
*Capacidad de ADSR		SI			NO		
Características video:							
*Resolución gráfica (PIXEL)	256x192	320x200	256x192	256x176	280x180	280x192	256x172
*Resolución multic (PIXEL)	64x48	160x200				40x43	
*Resolución textos	40x24	40x25	32x24	32x24	40x24	40x24	32x23
*Colores	16	16	16	8	15	6	8
*Sprites	32	16	16		—		—
Características interfaces:							
*Impresora paralela	Standard	Opcional	Opcional	Opcional	Opcional	Standard	Opcional
*Impresora serial	Opcional	Opcional	Opcional	Opcional	Opcional	Opcional	Opcional
*TV color PAL-N	Incorporada		Externa	Incorporada	NO	Incorporada	NO
*Salida monitor	SI	SI	SI	SI	SI	SI	SI
*Salida audio	SI	SI	SI	SI	SI	SI	SI
*Cassette audio (cable)	Standard	Opcional	Opcional	Standard	NO	Standard	Standard
*Joystick (conectores)	SI (2)	SI (2)	SI (2)	SI (2)	Opcional	SI (2)	SI (2)
*RS-232C	Opcional	Opcional	Opcional	Opcional	Opcional	Opcional	Opcional
*Para expansiones (SLOT)	SI (2)	SI (2)	SI (1)	NO	NO	NO	SI (1)
Accesorios opcionales:							
*Mouse	Opcional		NO		Opcional		NO
*Track Ball	Opcional		NO		NO		NO
*Lápiz óptico	Opcional						
*Lector código barras	Opcional						
*Tableta gráfica	Opcional		SI		Opcional		
*Modem c/asincrónico	Opcional				Opcional		
*Caja expansiones	Opcional		Opcional	NO	NO		
*Lector tarj. magnéticas	Opcional		NO	NO			
*Teclado numérico	Opcional		NO	NO	Opcional	NO	NO
Características archivos:							
*Grabador							
-Canales							
-Velocidad	1200/2400	1200		1500	3400		1500
-Cable p/grabador común	Standard	Opcional	Opcional		NO	Standard	Standard
*Disketera							
-Formato	5 1/4''	5 1/4''	5 1/4''		5 1/4''	5 1/4''	
-Capacidad almacenamiento	4x180 KB 4x360 KB	2x170 KB —	1x140 KB —	 —	6x140 KB —	2x140 KB —	 —
-Compatib. arch. MS-DOS	SI	NO	NO	NO	NO	NO	NO
-Velocidad							
Otras características							
*Fuente 220 V incorporada	SI	NO	NO	NO	NO	NO	NO
*Timer	SI	SI	SI	SI	NO	NO	SI
*Ranura p/cartridge soft	SI	SI	SI	NO	NO	NO	SI
*Expansión	SI	SI	NO	SI	SI	NO	NO



JUEGO DE LA OCA

CLASE: EDUCATIVO



Es un juego donde deberemos contestar sobre operaciones matemáticas.

Hay un tablero cuadriculado con escaleras y toboganes, y el objetivo es llegar a la salida.

Las escaleras nos permiten adelantar casilleros, pero si nos detenemos en alguno con tobogán, volveremos hacia atrás. Para avanzar, primero hay que contestar BIEN la cuenta que nos pregunte la computadora y segundo: lanzar un dado. Si nuestra respuesta es incorrecta o tardamos mucho tiempo para contestar, entonces perderemos el turno y en consecuencia no podremos tirar el dado. La computadora nunca pierde su turno.

El número que hayamos obtenido con el dato será la cantidad de cuadros que avanzaremos y si este número coincide con seis, podremos tirar dos veces el dado. La computadora no tiene esta ventaja porque ella tiene otra. La de no perder nunca el turno.

El juego está acompañado por sonido y es optativo agregarlo.

Al principio del juego se nos pregunta si queremos o no escuchar ruidos durante la ejecución del programa. Hay nueve niveles, la máquina graduará la dificultad de las preguntas. Esto significa que podremos subir o bajar de nivel dependiendo de nuestras respuestas. Las preguntas pueden ser sobre multiplicaciones o bien cuentas sobre las cuatro operaciones básicas (multiplicación, división, suma y resta) con cantidades sencillas o con números más complicados.

Este programa está realizado totalmente en BASIC y es sorprendente la rapidez con que se realizan los cambios de pantallas.

Pero aunque parezca que ganaremos, podemos llevarnos un chasco porque hast a el último momento no es seguro que el que se encuentre más avanzado sea el ganador.

Variables IMPORTANTES:

SO: sonido encendido o apagado
OP: opción del menú
DI: número de dificultad (es igual a la resta entre 10 y el dígito entrado)
PY(I): coordenada X para graficar en el tablero los muñecos en cualquiera de los 63 espacios
PY(I): coordenada Y
SO$, S1$: define los dos muñecos como sprites
PP, CP: posición del jugador y de la computadora sobre el tablero
TR: número de tentativas
OK: número de respuestas correctas
SI$: **operación matemática. ejemplo: suma, resta, etc.**
CA: respuesta correcta
PA$: respuesta del jugador
PT, CT: posición temporaria durante el chequeo
SC: tiro del dado
LI!: tiempo límite

Estructura del programa:

10- 100: pantalla de presentación.
110- 450: comentarios sobre el programa que saldrán por pantalla.
460- 450: cuerpo principal del programa.
650- 800: dimensiona y carga las coordenadas X e Y para cuadrilar el tablero.
820-1050: dibuja la cuadrícula de 9 casilleros por siete con los gráficos completos de los toboganes, las escaleras y el símbolo de salida del tablero.
1070-1320: organiza las operaciones matemáticas, llamando otras subrutinas. En esta sección se tiene en cuenta el límite de tiempo contenido en la variable DI.
1340-1370: sumas
1390-1420: restas
1440-1460: multiplicaciones
1480-1500: divisiones
1520-1580: tiro del dado
1600-1790: movida del jugador. En las líneas 1660 hasta la 1760 se fija si el sprite cae en algunos de los dibujos del tablero (tobogán, escalera o salida). Si esto sucede debemos ajustar las variables y realizar las operaciones correspondientes.
1810-1990: movida de la computadora con las verificaciones y operaciones similares a las anteriores.
2010-2040: mensajes si gana el jugador.
1060-2100: mensajes si gana la computadora.

```
10 GOSUB 650:CLS
20 COLOR,1,1:SCREEN 3:CLS:PRESET
(0,0): FOR T=32 TO 192 STEP32: RE
AD CH
30 C=RND(-TIME)*12+1:IF C=1 THEN
 30
40 COLOR C:PRESET (T,16)
50 PRINT#1,CHR$(CH);:FOR DL=1 TO
 200:NEXT:NEXT:PRESET (4,64):FOR
T= 32 TO 192 STEP 32:READ CH
60 C=RND(-TIME)*12+1:IF C=1 THEN
 60
70 COLORC:PRESET (T,64):PRINT#1,
CHR$(CH);:NEXT
80 FOR T=64 TO 160 STEP 32: READ
 CH:C=RND(1)*12+2:COLOR C:PRESET(
T,112)
90 PRINT#1,CHR$(CH);:FOR DL=1 TO
 800:NEXT:NEXT
100 DATA 74,85,69,71,79,32,68,69
,32,76,65,32,79,67,65
110 FOR DL=1 TO 5000:NEXT:COLOR
5,1,13: SCREEN 1: LOCATE 0,10: PR
INT"QUIERE EFECTOS SONOROS?":PRIN
T:PRINT"ENTRE S O N ";
120 IF INKEY$<>"" THEN 120
130 A$=INKEY$: IF A$<>"S" AND A$
<>"N" THEN 130
140 IF A$="S" THEN SO=-1 ELSE SO
=0
150 PRINT A$: FOR DL=1 TO 650:NE
XT:CLS:COLOR 12,1,1
160 LOCATE 3,3:PRINT"QUE TIPO DE
 PREGUNTAS ":PRINT:PRINT TAB(3);"
QUERES?"
170 LOCATE 2,9:PRINT"1 - TABLAS
DE MULTIPLICAR":PRINT:PRINT TAB(2
);"2 - TODAS LAS OPERACIONES"
180 PRINT TAB(6);"(NUMEROS FACIL
ES)": PRINT:PRINT TAB(2);"3 - TOD
AS LAS OPERACIONES"
190 PRINT TAB(6);"(NUMEROS DIFIC
ILES)"
200 IF INKEY$<>"" THEN 200
210 A$=INKEY$:IF VAL(A$)<1 OR VA
L(A$)>3 THEN 210
220 OP=VAL(A$):LOCATE 11,5:PRINT
 A$
230 FOR DL=1 TO 1000:NEXT:CLS:CO
LOR 5
240 PRINT TAB(3)"ELEGI EL NIVEL
DE DIFICUL-   TAD ? (1-9)":PRINT:
PRINT TAB(3)"1 ES DIFICIL - 9 ES
FACIL."
250 LOCATE 4,14:PRINT "PERO YO G
RADUARE EL NIVEL":PRINT TAB(4) "Q
UE VOS NECESITAS."
260 IF INKEY$<>"" THEN 260
270 A$=INKEY$:IF VAL(A$)<1 THEN
270
280 DI=VAL(A$):DI=10-DI:LOCATE 1
4,8:PRINT A$: FOR DL=1 TO 1000: N
EXT
290 SCREEN 0: COLOR 1,7: LOCATE
0,8:PRINT "EN ESTE JUEGO DEBES CO
NTESTAR"
300 PRINT "PREGUNTAS ARITMETICAS
.":PRINT :PRINT
310 PRINT "SI TE EQUIVOCAS O TAR
DAS MUCHO,":PRINT "PIERDES EL TUR
NO."
320 LOCATE 6,20:PRINT"PRESIONA L
A BARRA DE ESPACIO...": IF INKEY$
<>" " THEN 320
330 COLOR 14,1:SCREEN 2,2:SPRITE
$(0)=SO$:PUT SPRITE 0,(112,64),11
340 PRESET(144,72):PRINT#1,"VOS"
: SPRITE$(1)=S1$: PUT SPRITE 1,(1
12,88),5
350 PRESET(144,96):PRINT #1,"YO"
360 LINE(56,120)-(120,123),13,BF
:PRESET(144,120):PRINT #1,"TOBOGA
N"
370 LINE(56,142)-(120,142),3:LIN
E(56,144)-(120,144),3
380 PRESET(144,136):PRINT#1,"ESC
ALERA":FOR DL=1 TO7000:NEXT
390 COLOR 1,11:SCREEN 0: LOCATE
0,2:PRINT"PRESIONA SPACE PARA LAN
ZAR LOS DADOS.":PRINT
400 PRINT:PRINT "PARA SALIR DEL
TABLERO TENDRAS QUE": PRINT:PRINT
 "OBTENER EN LA ULTIMA TIRADA DE
DADOS,"
410 PRINT "EL NUMERO EXACTO DE C
ASILLEROS QUE":PRINT:PRINT"TE FAL
TAN PARA SALIR.": PRINT
420 PRINT:PRINT "TE PERMITIRE QU
E TIRES DENUEVO LOS":PRINT:PRINT"
DADOS SI SACAS UN SEIS"
430 PRINT:PRINT:PRINT:PRINT"JUGE
MOS...":PRINT:PRINT" PRESIONE LA
BARRA DE ESPACIO..."
440 IF INKEY$<>"" THEN 440
450 IF INKEY$<>" " THEN 450
460 REM
470 CLS:COLOR10,1:LOCATE 7,22:PR
INT"NIVEL DE DIFICULTAD =";10-DI
480 GOSUB 1070
490 IF NOT GO THEN LOCATE 12,20:
PRINT"PERDIO UN TURNO":FOR DL =1
TO 5000:NEXT
500 CLS: IF NOT GO THEN 560
510 LOCATE 14,12:PRINT"TU TURNO"
: GOSUB 1520
520 GOSUB 1600
530 IF PP=63 THEN 600
540 IF SC=6 THEN LOCATE 14,11:PR
INT"TIRA OTRA VEZ": IF SO THEN PL
AY"S11M900V6ECEC"
550 IF SC=6 THEN GOSUB 1520: GOT
O 520
560 FOR DL=1 TO 1500:NEXT: COLOR
 5: CLS: LOCATE 14,12: PRINT"MI T
URNO": GOSUB 1520
570 GOSUB 1810
580 IF CP=63 THEN 600
590 FOR DL= 1 TO 1500: NEXT: GOT
O 470
600 IF PP=63 THEN GOSUB 2010 ELS
E IF CP=63 THEN GOSUB 2060
610 FOR DL=1 TO 8000: NEXT: CLS:
 LOCATE 2,7: PRINT"QUIERE VOLVER
A JUGAR?(S/N) "
620 A$=INKEY$: IF A$<>"S" AND A$
<>"N" THEN 620
630 IF A$="S" THEN RUN ELSE CLS:
 END
640 '===========================
=====
650 SCREEN 0:DEFINT A-Z: KEY OFF
660 OPEN"GRP:" AS#1: PP=1: CP=1
670 DIM PX(64), PY(64): AX=8: AY
=160:ZI=1
680 FOR J=1 TO 4: FOR I=ZI TO ZI
+8
690 PX(I)=AX: PY(I)=AY:AX=AX+24:
NEXT
700 ZI=ZI+18: AX=8: AY=AY-48:NEX
```




```
T:ZI=18:AX=200:AY=136
710 FOR J=1 TO 3: FOR I=ZI TO ZI
+8
720 PX(I)=AX:PY(I)=AY:AX=AX-24:N
EXT
730 ZI=ZI+18:AX=200:AY=AY-48:NEX
T
740 M=2056:FOR J=1 TO 2: FOR I=1
 TO 8:B=VPEEK(M)
750 SO$=SO$+CHR$(B):M=M+1: NEXT
760 M=2056:NEXT: M=2064
770 FOR J=1 TO 2: FOR I=1 TO 8:B
=VPEEK(M)
780 S1$=S1$+CHR$(B):M=M+1: NEXT
790 M=2064:NEXT
800 SOUND 6,1: SOUND 7,46: RETUR
N
810 '===========================
=====
820 SCREEN 2,2: COLOR 9,1,1: CLS
830 FOR Y=12 TO 180 STEP24:LINE(
4,Y)-(220,Y): NEXT
840 FOR X=4 TO 220 STEP 24: LINE
 (X,12)-(X,180): NEXT
850 LINE(112,120)-(160,72),3:LIN
E(110,118)-(158,70),3
860 LINE(110,166)-(158,118),3:LI
NE(112,168)-(160,120),3
870 LINE(64,96)-(16,48),3:LINE(6
6,94)-(18,46),3
880 LINE(14,120)-(14,72),3:LINE(
16,120)-(16,72),3
890 LINE(112,96)-(88,72),3:LINE(
110,98)-(86,74),3
900 LINE(64,48)-(40,24),3:LINE(6
6,46)-(42,22),3
910 LINE(136,72)-(112,48),3:LINE
(138,70)-(114,46),3
920 LINE(184,96)-(208,72),13:LIN
E(183,95)-(207,71),13
930 LINE(182,94)-(206,70),13
940 LINE(207,144)-(207,96),13:LI
NE(208,144)-(208,96),13
950 LINE(209,144)-(209,96),13
960 LINE(87,48)-(87,24),13:LINE(
88,48)-(88,24),13
970 LINE(89,48)-(89,24),13
980 LINE(63,168)-(63,120),13:LIN
E(64,168)-(64,120),13
990 LINE(65,168)-(65,120),13
1000 PRESET(200,16):COLOR 15: PR
INT #1,"**": PRESET(200,24):PRINT
#1,"**"
1010 Y=168: FOR J=1 TO 2: FOR I=
1 TO 3:PRESET(224,Y):PRINT#1,"^"
1020 Y=Y-48:NEXT :Y=160: NEXT
1030 SPRITE$(0)=SO$: PUT SPRITE
0,(PX(PP),PY(PP)),11
1040 SPRITE$(1)=S1$: PUT SPRITE
1,(PX(CP)+8,PY(CP)),5
1050 RETURN
1060 '==========================
=====
1070 LOCATE 7,22:PRINT"NIVEL DE
DIFICULTAD= ";10-DI:GO=0
1080 IF OP=1 THEN GOSUB 1440: GO
TO 1100
1090 RR=RND(-TIME)*4+1: ON RR GO
SUB 1340,1390,1440,1480
1100 LOCATE 7,5:PRINT"QUE ES ";N
1;SI$;N2
1110 PRINT :PRINT TAB(7)"RESPUES
TA =";:PA$="":TIME=0
1120 IF DI<4 THEN LI!=875/DI ELS
E LI!=32*DI+378
1130 AA$=INKEY$: IF AA$<>"" THEN
 1130
1140 ZZ$=INKEY$: IF ZZ$="" THEN
1140
1150 IF TIME>LI! THEN TR=TR+1: G
OTO 1210
1160 PRINT ZZ$;
1170 PA$=PA$+ZZ$: IF LEN(PA$)< L
EN(STR$(CA))-1 THEN 1140
1180 TR=TR+1
1190 IF VAL(PA$)=CA THEN GO=-1:
OK=OK+1: LOCATE 7,10: PRINT"CORRE
CTO": GOTO 1220
1200 LOCATE 7,1: PRINT"MAL!! - L
A RESPUESTA CORRECTA         ES ";
CA: GOTO 1220TA OK  O
1210 LOCATE 7,2:PRINT"DEMASIADO
LENTO!"
1220 FOR DL=1 TO 3000: NEXT
1230 IF TR=3 AND OK<2 AND SO THE
N DI=DI-1:PLAY"S8M200007EEEE": GO
TO 1290
1240 IF TR=3 AND OK<2 THEN DI=DI
-1: GOTO 1290
1250 IF TR=3 AND OK=3 AND SO THE
N DI=DI+1:PLAY"S8M200007EEEE": GO
TO 1290
1260 IF TR=3 AND OK=3 THEN DI=DI
+1: GOTO 1290
1270 IF TR=3 THEN TR=0: OK=0: RE
TURN
1280 RETURN
1290 TR=0:OK=0:IF DI<1 THEN DI=1
 ELSE IF DI>9 THEN DI=9
1300 LOCATE 7,22:PRINT"NIVEL DE
DIFICULTAD =":ZZ=10-DI:FOR I=1 TO
 6: LOCATE 27,22
1310 PRINT" ": FOR DL=1 TO 250:
NEXT: LOCATE 26,22:PRINT ZZ: FOR
DL=1 TO 500: NEXT: NEXT
1320 RETURN
1330 '==========================
=====
1340 IF OP=2 THEN N1=RND(-TIME)*
10+1: N2=RND(-TIME)*10+1: GOTO 13
60
1350 N1=RND(-TIME)*99+1: N2=RND(
1)*99+1
1360 SI$=" MAS ": CA=N1+N2
1370 RETURN
1380 '==========================
=====
1390 IF OP=2 THEN N1=RND(-TIME)*
10+1: N2=RND(5)*10+1:IF N2>=N1 TH
EN 1390 ELSE GOTO 1410
1400 N1=RND(-TIME)*99+1:N2=RND(1
)*99+1: IF N2>=N1 THEN 1400
1410 SI$=" MENOS": CA=N1-N2
1420 RETURN
1430 '==========================
=====
1440 N1=RND(-TIME)*12+1:N2=RND(6
)*12+1
1450 SI$=" VECES ":CA=N1*N2
1460 RETURN
1470 '==========================
=====
1480 N2=RND(9)*12+1: IF OP=2 THE
N CA=RND(5)*5+1 ELSE CA=RND(7)*20
+1
1490 SI$=" "+CHR$(246)+" ":N1=N2
*CA
1500 RETURN
1510 '==========================
=====
1520 LOCATE 16,15: PRINT STRING$
(5,CHR$(219))
1530 LOCATE 16,19: PRINT STRING$
(5,CHR$(219))
1540 FOR I=1 TO 3: LOCATE 16,15+
I: PRINT CHR$(219) SPC(3) CHR$(21
9): NEXT
1550 LOCATE 7,6: PRINT"PRESIONA
LA BARRA DE ESPACIO.": IF INKEY$<
>"" THEN 1550
1560 IF INKEY$<>" " THEN 1560
1570 FOR I=1 TO 40:SC=INT(RND(-T
IME)*6+1): LOCATE 17,17: PRINT SC
;:NEXT
1580 FOR DL=1 TO 2000: NEXT: SCR
EENO: RETURN
1590 '==========================
=====
1600 IF PP+SC>63 AND SO THEN PLA
Y"O2V10EC"
1610 IF PP+SC>63 THEN LOCATE 15,
12:PRINT "DEMASIADO GRANDE":RETUR
N
1620 GOSUB 820: PUT SPRITE 0,(PX
(PP),PY(PP)),11
1630 FOR DL=1 TO 1000:NEXT:FOR I
=1 TO SC: PP=PP+1
1640 PUT SPRITEO,(PX(PP),PY(PP))
,11:IF SO THEN BEEP
1650 FOR DL=1 TO 1000: NEXT: NEX
T:PT=PP
1660 RESTORE 2120:FOR I=1 TO 11:
 READ AA,BB: IF PP=AA THEN PP=BB
1670 NEXT
1680 IF PP<PT THEN 1740 ELSE IF
PP=PT THEN 1780
1690 XX=PX(PT):FOR I=PY(PT) TO PY
(PP) STEP-8: PUT SPRITE 0,(XX,I),
11
1700 IF PT=5 OR PT=23 THEN XX=XX
+8
1710 IF PT=32 OR PT=34 OR PT=42
OR PT=52 THEN XX=XX-8
1720 NEXT:IF SO THEN SOUND 8,15:
 SOUND 8,0: SOUND 9,0
1730 GOTO 1780
1740 XX=PX(PT):FOR I=PY(PT) TO P
Y(PP) STEP 8: PUT SPRITE 0,(XX,I)
,11
1750 IF PT=45 THEN XX=XX-8
1760 NEXT: IF SO THEN SOUND 8,15
: SOUND 9,15: FOR J=0 TO 1: FOR I
=1 TO 250 STEP 16
1770 IF SO THEN SOUND 0,I: SOUND
 1,J:NEXT: NEXT: SOUND 8,0: SOUND
 9,0
1780 IF PP=63 AND SO THEN PLAY"M
500006ECECECEC"
1790 FOR DL=1 TO 3000: NEXT: CLS
: SCREEN 0: RETURN
1800 '==========================
=====
1810 IF CP+SC>63 AND SO THEN PLA
Y"O2V10EC"
1820 IF CP+SC>63 THEN LOCATE 15,
12:PRINT"DEMASIADO GRANDE":RETURN
1830 GOSUB 820: PUT SPRITE 1,(PX
(CP)+8,PY(CP)),5
1840 FOR DL=1 TO 1000: NEXT: FOR
 I=1 TO SC: CP=CP+1
1850 PUT SPRITE 1,(PX(CP)+8,PY(C
P)),5: IF SO THEN BEEP
1860 FOR DL=1 TO 1000: NEXT:NEXT
:CT=CP
1870 RESTORE 2120: FOR I=1 TO 11
: READ AA,BB: IF CP=AA THEN CP=BB
1880 NEXT
1890 IF CP<CT THEN 1950 ELSE IF
CP=CT THEN 1980
1900 XX=PX(CT)+8: FOR I=PY(CT) T
O PY(CP) STEP-8: PUT SPRITE 1,(XX
,I),5
1910 IF CT=5 OR CT=23 THEN XX=XX
+8
1920 IF CT=32 OR CT=34 OR CT=42
OR CT=52 THEN XX=XX-8
1930 NEXT: IF SO THEN SOUND 8,15
: SOUND 9,15: FOR J=1 TO 0 STEP-1
: FOR I=250 TO 1 STEP -16: SOUND
0,I: SOUND 1,J: NEXT: NEXT: SOUND
 8,0: SOUND 9,0
1940 GOTO 1980
1950 XX=PX(CT)+8: FOR I=PY(CT) T
O PY(CP) STEP 8: PUT SPRITE 1,(XX
,I),5
1960 IF CT=45 THEN XX=XX-8
1970 NEXT: IF SO THEN SOUND 8,15
: SOUND 9,15: FOR J=0 TO 1: FOR I
=1 TO 250 STEP 16: SOUND 0,I: SOU
ND 1,J: NEXT :NEXT: SOUND 8,0: SO
UND 9,0
1980 IF CP=63 AND SO THEN PLAY"M
500006ECECECEC"
1990 FOR DL=1 TO 3000: NEXT: CLS
: SCREENO: RETURN
2000 '==========================
=====
2010 COLOR 10: LOCATE 13,7: PRIN
T"USTED GANA"
2020 IF CP>55 THEN LOCATE 7,12:
PRINT"PERO RASPANDO....": GOTO 21
00
2030 IF CP<39 THEN LOCATE 7,12:
PRINT"YO ESTUVE BASTANTE ATRAS.":
 GOTO 2100           USTED GANA
2040 LOCATE 12,15: PRINT "BIEN H
ECHO": GOTO 2100
2050 '==========================
=====
2060 LOCATE 13,7: PRINT"YO GANE!
!!"
2070 IF PP>55 THEN LOCATE 7,12:
PRINT"PERO ME PISASTE LOS TALONES
": GOTO 2090
2080 IF PP<39 THEN LOCATE 7,12:
PRINT "QUE LEJOS QUE ESTABAS!!!":
 GOTO 2090
2090 LOCATE 12,15: PRINT"MALA SU
ERTE!!"
2100 LOCATE 3,18: PRINT"TU ULTIM
O NIVEL FUE: ";10-DI
2110 RETURN
2120 DATA 5,25,19,37,21,3,23,43,
28,10,32,40,34,54,42,50,45,29,52,
56,58,51
```

CAJERO

COMERCIAL

Insistiendo en tratar de demostrar que las computadoras no se inventaron sólo para que las disfrutemos jugando, aquí tenemos un programa que nos ayudará a llevar la caja. Si, por ejemplo, un colegio realiza un espectáculo para los alumnos, padres y profesores y el valor de la entrada varía de acuerdo a los distintos niveles en que se encuentra el alumno (secundario, primario y jardín de infantes) y además se diferencia entre profesores y padres, entonces este programa es ideal para facilitarnos el "doloroso" trabajo de recolectar el dinero.

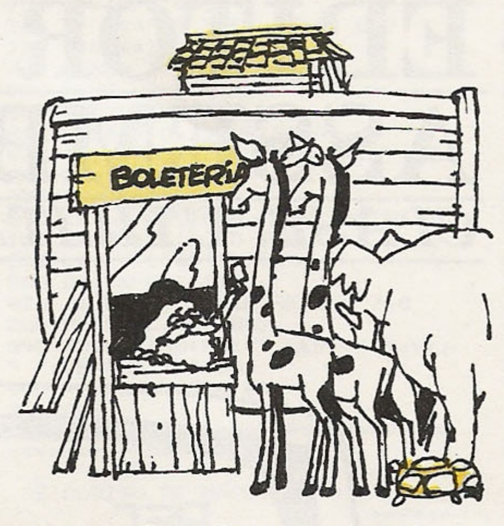
Pero obviamente el programa se puede ajustar a cada necesidad específica como controlar la caja de un cine o exposición.

Y para las MSX que tengan conectada una impresora, saldrá el resúmen de las entradas vendidas.

La entrada de datos se efectúa presionando las teclas de función.

- **1** — padres
- **2** — secundario
- **3** — primario
- **4** — jardín de infantes
- **6** — cancela la entrada de datos
- **10** — imprime los resultados del día

Para imprimir la cantidad de personas que ingresaron a ver el espectáculo, pero sin detallar el dinero recolectado, una vez que se terminó de ingresar los datos, presionar RETURN.



También nos puede decir cual es el vuelto que debemos dar.

Pero si no tenemos impresora y queremos que el procesamiento de los datos se muestre por pantalla, podemos reemplazar en el programa las instrucciones LPRINT por PRINT. Como en la línea 430, que al cambiar las instrucciones que mencionamos antes será: 430 PRINT "TOTALES DEL DIA"

Los valores de las entradas que tiene el programa están en centavos, pero se pueden modificar. En la línea 50, A% es el valor para los padres, c% para los alumnos del secundario, o% para primario y p% para jardín de infantes. Nosotros hemos cargado estas variables con 20,12,10,10 respectivamente.

Otra opción sería agregar a la salida por impresora el nombre del lugar donde estamos contabilizando.

Para esto agreguemos la línea 685 LPRINT: LPRINT "COLEGIO..."

Variables IMPORTANTES:

- **TA%** número de padres
- **TC%** número de alumnos del secundario
- **TS%** número de alumnos del primario
- **TP%** número de alumnos de jardín de infantes
- **TT%** total de dinero recolectado
- **O%** valor de la entrada para primario
- **A%** valor para los padres
- **c%** entrada para los del secundario
- **p%** valor de la entrada para los del jardín

Estructura del programa:

- **10- 30:** selecciona modo y color
- **50:** inicializa valores de las entradas
- **60:** inicializa variables
- **70-160:** inicializa teclas de función
- **170-210:** loop principal
- **220-360:** menú con las posibles entradas
- **370-410:** verifica que la entrada de los datos sea correcta
- **420-550:** salida de los datos entrados durante el día
- **560-600:** calcula el vuelto, mensaje para continuar la siguiente operación
- **610-660:** actualiza la cantidad de entradas vendidas
- **670-750:** imprime la concurrencia

```
10 KEY OFF
20 SCREEN 0
30 COLOR 15,1,15
40 REM ******
50 A%=20:C%=12:O%=10:P%=10
60 NA%=0:NC%=0:NO%=0:NP%=0
70 CLS
80 KEY 1,"A"
90 KEY 2,"c"
100 KEY 3,"s"
110 KEY 4,"p"
120 KEY 5,"t"
130 KEY 6,"q"
140 KEY 7,"m"
150 KEY 8,"y"
160 KEY 9,"n":KEY 10,"e"
170 GOSUB 220
180 GOSUB 560
190 GOSUB 680
200 GOTO 70
210 END
220 CLS:TA%=0:TC%=0:TS%=0:TP%=0
230 A$=INKEY$
240 IF A$="a" OR A$="A" THEN TA%
=TA%+1
250 IF A$="c" OR A$="C" THEN TC%
=TC%+1
260 IF A$="s" OR A$="S" THEN TS%
=TS%+1
270 IF A$="p" OR A$="P" THEN TP%
=TP%+1
280 IF A$="e" OR A$="E" THEN GOS
UB 620: GOSUB 430
290 IF A$="q" OR A$="Q" THEN GOT
O 220
300 LOCATE 2,5:PRINT TA%;" Padre
s          ...";TA%*A%;" c"
310 LOCATE 2,7:PRINT TC%;" Secun
dario      ...";TC%*C%;" c"
320 LOCATE 2,9:PRINT TS%;" Prima
rio        ...";TS%*O%;" c"
330 LOCATE 2,11:PRINT TP%;" Jard
ín de infan ...";TP%*P%;" c"
340 TT%=TA%*A%+TC%*C%+TS%*O%+TP%
*P%
350 LOCATE 5,15:PRINT"TOTAL ....
...........   ";TT%/100
360 IF A$<>CHR$(13) THEN GOTO 23
0
370 REM ****
380 LOCATE 1,18:PRINT"Es esto co
rrecto (S/N)?
390 A$=INKEY$:IF A$="N" OR A$="n
" THEN 220
400 IF A$="S" OR A$="s" THEN RET
URN
410 GOTO 390
420 REM ********
430 LPRINT "********** TOTALES DE
L DIA *********"
440 LPRINT "Tipo","número","vent
as"
450 LPRINT "Padres ",NA%,NA%*A%/
100
460 LPRINT "Secundario",NC%,NC%*
C%/100
470 LPRINT "Primario",NO%,NO%*O%
/100
480 LPRINT "J/de infan.",NP%,NP%
*P%/100
490 TN%=NA%+NC%+NO%
500 TM%=(NA%*A%)+(NC%*C%)+(NO%*O
%)+(NP%*P%)
510 LPRINT "TOTALES :- ",TN%,TM%
/100
520 CLS:PRINT"Quiere volver al p
rograma <S/N> ?"
530 AA$=INKEY$: IF AA$="S" OR AA
$="s" THEN RUN
540 IF AA$="N" OR AA$="n" THEN C
LS: PRINT "Adios...":END
550 GOTO 530
560 CLS: LOCATE 2,2: INPUT"SE PA
GO CON:- ";CA
570 IF CA<TT%/100 THEN GOTO 560
580 PRINT: PRINT: PRINT: PRINT"H
ay que devolver ";CA-TT%/100
590 LOCATE 1,10:PRINT "Presione
la barra espaciadora cuando este
listo para la próxima operación"
600 A$=INKEY$: IF A$<>" " THEN 6
00
610 REM *****
620 NA%=NA%+TA%
630 NC%=NC%+TC%
640 NO%=NO%+TS%
650 NP%=NP%+TP%
660 RETURN
670 REM *******
680 LPRINT :LPRINT" Concurrencia
"
690 LPRINT:LPRINT "*************
*************************"
700 IF TA%>0 THEN LPRINT TA%;" P
adres"
710 IF TC%>0 THEN LPRINT TC%;" S
ecundario"
720 IF TS%>0 THEN LPRINT TS%;" P
rimario"
730 IF TP%>0 THEN LPRINT TP%;" J
ardín de infan."
740 LPRINT "*********************
*****************"
750 RETURN
```



EDITOR ASSEMBLER

CLASE: UTILITARIO

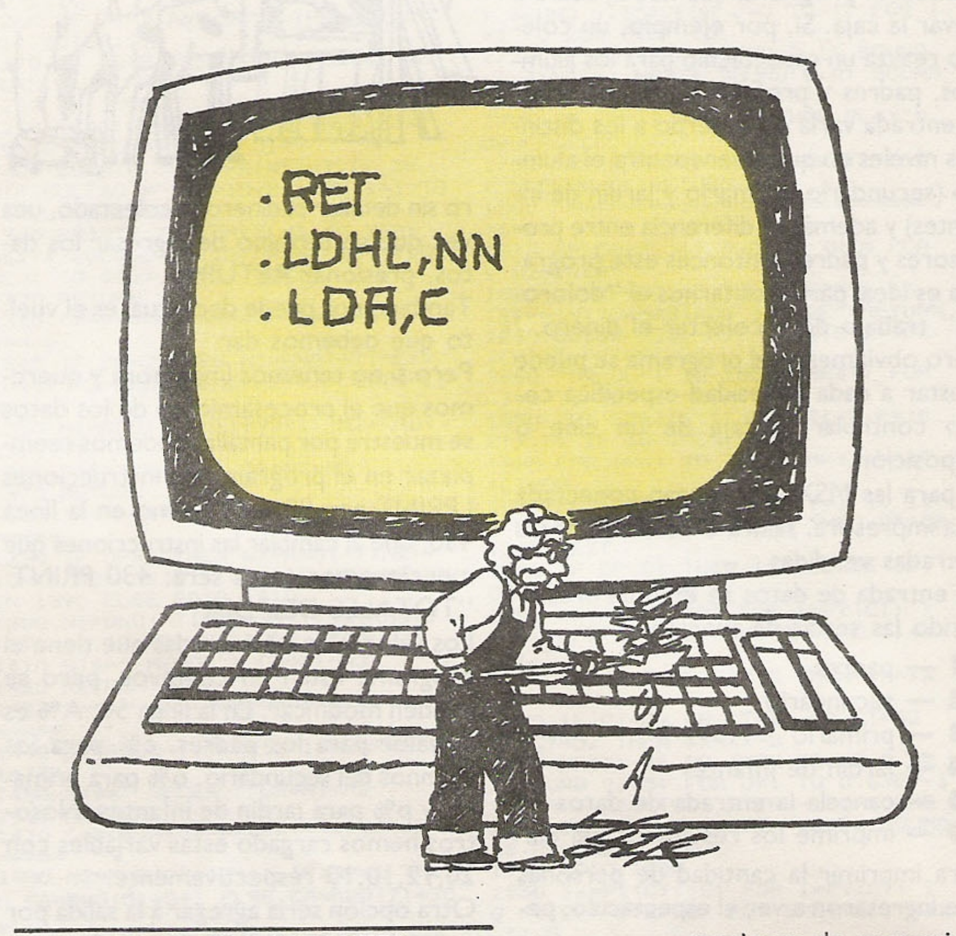


Para escribir datos en Assembler en la memoria de la computadora, existen diversos métodos y este es uno.

En el ambiente de los programadores, este programa es un editor Z80, pues es utilizado para programar el microprocesador Z80 —cerebro de las MSX—. Este editor permite leer, escribir, grabar y cargar rutinas en código de máquina.

Cuando consideramos terminado el ingreso de datos, presionemos la tecla de FUNCTiON 5 seguida por RETURN. Aparecerá el mensaje: "Ejecución abortada..." y nos marcará, en el renglón inferior, en que modo nos hallamos. Cuando estemos en el modo "CTRL" podemos seleccionar la próxima operación que realizará la computadora, pudiéndose escoger entre: leer direcciones de memoria (F1), grabar en cinta (F2), cargar desde cinta (F3) o escribir código de máquina en la memoria (F5). Las direcciones, como los datos, deberán ser entrados en hexadecimal.

Para almacenar las rutinas en Assembler que construyamos con este editor, podemos también usar la disketera, modificando las líneas: 550 y 600. La primer línea se podrá modificar de la siguiente forma: 550 BSAVE"Z-80", &HDOOO, MX y la segunda por: 600 BLOAD"Z-80". Pero de esta manera, cada subrutina en Assembler será cargada con el mismo nombre. La línea 510 se puede borrar si trabajamos con la disketera y no con el cassette.



Variables IMPORTANTES

PC: Dirección de memoria inicial
FA: Dirección de memoria final para la lectura
MX: Ultima dirección donde escribimos

Estructura del programa:

10- 160: Inicialización de variables.
170- 230: Prepara teclas de función.
240- 480: Rutina que permite leer direcciones de memoria.
490- 570: Rutina para grabar en cinta.
580- 620: Carga desde cinta, código de máquina.
630- 640: Corta la actividad que estaba realizando y vuelve a aceptar un modo.
650- 900: Acepta código de máquina.
910- 970: Pantalla del comienzo.
980-1260: Operaciones y datos.

```
10 KEY OFF
20 DEFINT A-Z
30 CLEAR 8192,&HD000
40 GOSUB 910
50 PRINT "ESPERE..... POR FAVOR"
60 DIM L(255),X(255)
70 DA=0:PC=127:MX=&HD000
80 READ D$,FI$
90 W1=1
100 FOR P=DA TO PC
110 L(P)=VAL(MID$(D$,W1,1))
120 X(P)=VAL(MID$(FI$,W1,1))
130 W1=W1+1
140 NEXT P
150 DA=DA+128:PC=PC+128
160 IF DA=256 THEN GOTO 170 ELSE
 GOTO 80
170 LOCATE 0,CSRLIN-1:PRINT"LIST
O....MODO CONTROL  "
180 GOSUB 1210
190 KEY ON
200 KEY(1) ON:KEY(2) ON:KEY(3) O
N: KEY(4) ON:KEY(5) ON
210 ON KEY GOSUB 240,490,580,630
,650
220 KEY4,">CTRL<"
230 GOTO 180
240 GOSUB 1210
250 KEY1,">LEER<"
260 PRINT :PRINT "ENTRE LA DIREC
CION DE COMIENZO"
270 INPUT PC
280 IF PC<0 THEN PC=65536!+PC
290 PRINT "ENTRE LA ULTIMA DIREC
CION (OPCIONAL)"
300 INPUT FA
310 IF FA=0 THEN FA=&HFFFF
320 IF FA<0 THEN FA=65536!+FA
330 PRINT"ENTRE EL CODIGO A BUSC
AR            (H PARA EXPLICACIO
N DE AYUDA)          ( OPCIONAL)"
340 FI$="":INPUT FI$
350 IF LEFT$(FI$,1)="H" THEN GOS
UB 460
360 IF FI$<>"" THEN FI$=FI$+" "
ELSE LET FI$="XX"
370 GOSUB 980
380 PC=PC+LE
390 PRINT A$
400 IF MID$(A$,7,LEN(FI$))=FI$ T
HEN GOSUB 440
410 IF PC<FA THEN GOTO 370
420 PRINT" FIN DEL LISTADO"
430 RETURN 210
440 PRINT:PRINT:LOCATE 20,19:PRI
NT"ENCONTRADO"
450 IF INKEY$<>"" THEN RETURN:FI
=0:PRINT ELSE GOTO 450
460 PRINT:PRINT" USTED PUEDE PED
IRLE A LA COMPUTADORA BUSCAR EN E
L LISTADO HASTA QUE EN-   CUENTRE
 EL CODIGO DE MAQUINA QUE     UST
ED ESPECIFICO. ESTO PODRA DETE-
 NERSE CON PRESIONAR CUALQUIER TE
```

```
CLA "
465 PRINT"ESTO LE PERMITIRA HACE
R LAS ANOTACIO-NES QUE USTED DESE
A"
470 PRINT "EL CODIGO QUE QUIERE
BUSCAR DEBE SER ENTRADO EN LA FOR
MA 21 00 D3(RETURN).LA COMPUTADOR
A PUEDE ACEPTAR PARTE   DEL CODIG
O e.g. 21(RETURN) Y PUEDE   PARAR
 EN CUALQUIER COMIENZO DE INS-  T
RUCCION COMENZANDO CON 21.
   TODOS LOS CODIGOS DE"
475 PRINT "INTRUCCIONES DEBEN EN
TRAR EN HEXA."
480 PRINT:RETURN 330
490 GOSUB 1210:KEY 2,">GRABAR<"
500 TIME=0
510 MOTOR ON
520 PRINT "PRECIONE -RECORD- EN
SU GRABADOR"
530 IF TIME<200 THEN GOTO 530
540 PRINT"AHORA GRAVANDO..."
550 BSAVE"CAS:Z-80",&HD000,MX
560 PRINT"TERNIME..."
570 RETURN
580 GOSUB 1210:KEY 3,">CARGAR<"
590 PRINT "COMIENZE LA GRABACION
..."
600 BLOAD"CAS:"
610 PRINT"HECHO..."
620 RETURN
630 PRINT"EJECUCION ABORTADA..."
640 RETURN 210
650 GOSUB 1210
660 KEY5,">ESCRIBIR<"
665 PRINT"PARA ESCAPAR DEL MODO
ESCRIBIR,       DEBERA PRESIONAR C
TRL(RETURN)"
670 PRINT,,"ENTRE LA DIRECCION D
EL COMIENZO ";
680 INPUT PC
690 PRINT: PRINT "LISTO PARA ING
RESAR LOS DATOS-"
700 PRINT:PRINT:PRINT
710 CR=0
720 LO=PC
730 IF LO>MX THEN MX=LO
740 IF LO>=&HF000 OR LO<&HD000 T
HEN PRINT "USTED TRATA DE ESCRIBI
R FUERA DEL    AREA DESPEJADA":GO
TO 950
750 D$=INPUT$(1)
760 IF D$="*" THEN GOSUB 860
770 D$=D$+INPUT$(1)
780 GOSUB 1150
790 POKE LO,DA
800 GOSUB 980
810 LOCATE 0,20:PRINT A$
820 IF LE<>CR+1 THEN LOCATE (LO-
PC)*3+11,20:PRINT"<";
830 CR=CR+1
840 IF LE=CR THEN PRINT:PRINT:PR
INT:PC=PC+LE:GOTO 710
850 LO=LO+1:GOTO 750
860 D$=INPUT$(1)
870 IF D$="*" THEN RETURN 710
880 POKE PC,ASC(D$)
890 GOSUB 980:PRINT LEFT$(A$,8);
" ";:PRINT"  ";D$
900 PC=PC+1:GOTO 860
910 CLS
920 PRINT"*************************
****************
                 ** Z80 CARGADOR
 DE CODIGO DE MAQUINA **
                          *******
*********************************"
930 PRINT:PRINT:PRINT "USE LAS I
ECLAS DE FUNCION PARA       CAMBI
AR EL MODO. ASEGURESE DE ESCRI- B
IR CON MAYUSCULAS":PRINT:PRINT
940 PRINT"SAVE Y LOAD SON PARA G
RAVAR EL CODIGODE MAQUINA SOLAMEN
TE Y NO PARA LA    PARTE DE BASIC
.":PRINT:PRINT
950 PRINT"EL ESPACIO PARA CODIGO
 DE MAQUINA    RESERVADO VA DESDE
&HD000 AL &HEFFF. EN TOTAL 8K DE
 CODIGO DE MAQUINA."
960 PRINT
970 RETURN
980 B1=PEEK(PC)
990 B2=PEEK(PC+1)
1000 LET A$=RIGHT$("000"+HEX$(PC
),4)+"  "
1010 IF HEX$(B1)="DD"OR HEX$(B1)
="FD" THEN GOSUB 1080
1020 IF HEX$(B1)="ED" THEN GOSUB
 1090
1030 LE=L(B1)
1040 FOR P=LE TO 1 STEP -1
1050 A$=A$+RIGHT$("0"+HEX$(PEEK(
PC+LE-P))+" ",3)
1060 NEXT P
1070 RETURN
1080 L(B1)=X(B2):RETURN
1090 F$=LEFT$(HEX$(B2),1)
1100 S$=RIGHT$(HEX$(B2),1)
1110 IF F$="4" OR F$="5" OR F$="
6" OR F$="7" THEN GOTO 1130 ELSE
L(B1)=2:RETURN
1120 IF F$="4" OR F$="5" OR F$="
6" OR F$="7" THEN GOTO 1130 ELSE
L(B1)=2:RETURN
1130 IF S$="3" OR S$="B" THEN L(
B1)=4:RETURN ELSE L(B1)=2:RETURN
1150 W1=ASC(LEFT$(D$,1))
1160 W2=ASC(RIGHT$(D$,1))
1170 IF W2>64 THEN W2=W2-7
1180 IF W1>64 THEN W1=W1-7
1190 DA=W1*16+W2-816
1200 RETURN
1210 KEY1," READ ":KEY 2," SAVE
":KEY 3," LOAD ":KEY 4," CTRL ":K
EY 5," WRITE ":KEY 6,"":KEY 7,"":
KEY 8,"":KEY 9,"":KEY 10,""
1220 RETURN
1230 DATA "131111211111112123111
12121111121233111212131112123311
21213111211111111111111111111111
11111111111111111111111111111111
11111111"
1240 DATA "222222222222222222222
22222222222244222222242222222233
42222222223333333333333333333333
33333333333333333333333333333333
33333333"
1250 DATA "111111111111111111111
11111111111111111111111111111111
11111111111333121113233211132312
11132342111313121113134211131312
11313421"
1260 DATA "333333333333333333333
33333333333333333333333333333333
33333333332222222222242222222222
22222222222222222222222222222222
22222222"
```



PROCESADOR DE PALABRAS

El ingeniero C. Atashian y el señor A. Capo, gente de Talent Río Grande (Tierra del Fuego), a pesar de la distancia, no se olvidan de los usuarios de sus computadoras ni de esta publicación dedica-exclusivamente a ellos.

Es así como nos han enviado la siguiente carta, explicando su trucazo para transformar, con un par de líneas, a nuestra computadora en un procesador de textos.

Río Grande, 11 de junio 1986

Sres. LOAD MSX:

Para todos aquellos que no posean un wordprocessor o editor de textos les enviamos este programa, el cual les permite escribir cualquier cosa en pantalla (una carta, formulario, etc.) y volcarlo luego en impresora (screen-dump) presionando solamente F1.

Sin otro particular, saludamos a Uds. atte.

Ing. C. Atashian y Sr. A. Capo

La posdata aclara que esta carta fue escrita por medio de este artilugio.

```
10 KEY OFF: KEY 1,CHR$(30)+CHR$(
13):CLS: LOCATE 0,0:LINE INPUT A$
20 DEFINT A-Z: FOR F=0 TO 23: FO
R C=0 TO 39: LPRINT CHR$(VPEEK(F*
40+C));:NEXT: LPRINT CHR$(13): NE
XT: LOCATE 0,22
```

Inverse video

Los caracteres tienen un color de fondo y otro de frente o tinta. Con la instrucción COLOR 15,1 pedimos que el color de frente sea blanca y el de fondo negro; pero después de ejecutar el programa de la figura cinco, el color de frente será negro y el de fondo obviamente blanco. De manera que el programa invierte los colores de los caracteres.

Se puede cambiar todos los caracteres o sólo un par de ellos. Para seleccionar alguna de las dos formas, cuando el ordenador nos pregunta "CARACTER A CONVERTIR", debemos contestar con "TODOS" o presionar la tecla que queremos cambiar.

Si entramos "TODOS", no interrumpamos la ejecución del programa pensando que la máquina volvió a tomarse un descanso dejándonos sentados en el banquito de espera. Pasarán varios segundos hasta que haya solucionado la misión que le encomendamos. A medida que se produzcan las conversiones, los caracteres de la pantalla aparecerán cambiados.

Ahora podemos obtener un INVERSE VIDEO para todos los caracteres o sólo para una parte de ellos.

```
40 SCREEN 1
50 CLS
60 INPUT"CARACTER A CONVERTIR";A$
70 IF A$="TODOS" OR A$="todos" THEN Y=0:L=2048:TIME=0:GOTO 90
80 Y=ASC(A$)*8:L=Y+7
90 FOR I=Y TO L
100 IF I=256 THEN I=264
110 D$="0000000"
120 F$=RIGHT$(D$+BIN$(VPEEK(I)),8)
130 FOR B=1 TO 8
140 V$(B)=MID$(F$,B,1)
150 IF V$(B)="0" THEN V$(B)="1":GOTO 170
160 IF V$(B)="1" THEN V$(B)="0"
170 NEXT B
180 G$=V$(1)+V$(2)+V$(3)+V$(4)+V$(5)+V$(6)+V$(7)+V$(8)
190 VPOKE I,VAL("&B"+G$)
200 NEXT I
210 PRINT TIME:INPUT "OTRO( s/n)";S$
220 IF S$="S" OR S$="s" THEN 50 ELSE END
```

ALTA RESOLUCION

Aprendamos a aplicar todas las facilidades gráficas que tiene nuestra computadora y veamos algunos ejemplos.

Hay cuatro modos distintos de pantalla,dos para textos y dos para gráficos. Pero ¿para qué dos de cada uno y no menos?

Como dijimos, los cuatro tipos de pantalla son diferentes, el modo 0 es de texto y permite escribir 24 líneas de 37 caracteres cada una; mientras que el otro modo de texto, el uno, admite 24 líneas de 29 caracteres cada uno. Entonces notemos que la capacidad del modo 1 es menor que la del 0. Por eso decimos mos que el modo 1 es de baja resolución para textos.

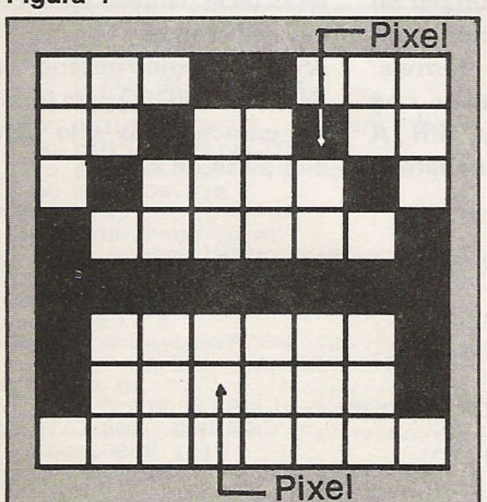
Y para gráficos, el modo 2 es de alta resolución porque se pueden ubicar 192 filas y cada una contiene 256 "puntitos".

El nombre "puntitos" no es el correcto, en realidad se denominan pixels. Un pixel es la unidad más pequeña de una pantalla de video. El término pixel corresponde a PIX de picture (dibujo, cuadro) y EL de element (elemento), o sea elemento de la imagen.

El último modo, el tres, corresponde a gráficos pero tiene la posibilidad de posicionar solamente 48 filas por 64 pixels cada una. Las filas en los modos gráficos también son pixels.

Observemos que para los modos de textos hablamos de capacidad de caracteres, mientras que en los gráficos destacamos la cantidad de pixels que entran por pantalla.

Figura 1



Cada caracter está formado por pixels, como se ve en la figura uno. Si nos encontramos trabajando en el modo 0, cada caracter se identifica por 5 pixels horizontales y 7 verticales. En cambio en el modo de texto 1, los caracteres están definidos por 8 pixels verticales y 8 horizontales.

Para graficar, es necesario trabajar en cualquiera de los modos dos o tres.

Para el tipo de alta resolución gráfica, la pantalla está cuadriculada como se muestra en la figura 2. La figura tres representa la división de video para el modo 3, de baja resolución gráfica.

A pesar que los modos dos y tres son para dibujar, también es posible escribir en ellos, pero no con la instrucción PRINT directamente.

Hay que agregar la sentencia OPEN "GRP:" AS#1 al principio del programa e incorporarles a las instrucciones PRINT el símbolo numeral seguido por el 1. Entonces las instrucciones para escribir serían de esta for: PRINT#1, "...".

Pero llegó el momento de ver detalladamente cada una de las instrucciones que nos permitirán dibujar.

Si queremos graficar en alta resolución (modo 2) hay que declararlo al principio del programa con la instrucción SCREEN seguido por el número del modo donde queremos trabajar. Por ejemplo SCREEN 2.

Figura 2

0 1 2 3 .... 253 254 255

0
1
2
3
...
189
190
191

PSET

Esta sentencia es simple porque tiene la forma: PSET (X,Y), color. Esta orden pinta el pixel ubicado en la columna X, fila Y —ver figura 4 para ver cuáles son las filas y cuáles las columnas—.

El parámetro color no es obligatorio especificarlo. En este caso la compuadora —el sistema— pintará el pixel con el último color declarado.

El punto a partir del cual se cuentan las posiciones, es el vértice superior izquierdo. Entonces este vértice es el que tiene valor cero para X y cero para Y.

Si entramos la instrucción PSET (150,150),8 visualizaremos un punto pintado de rojo en la columna 150, fila 150. Y si entramos PSET (2,4) veremos el pixel de la segunda columna, cuarta fila, pintado también de rojo porque como dijimos antes, utiliza el último color declarado.

En cada uno de los casos que vimos anteriormente, la computadora utiliza como punto de referencia el (0,0), o sea

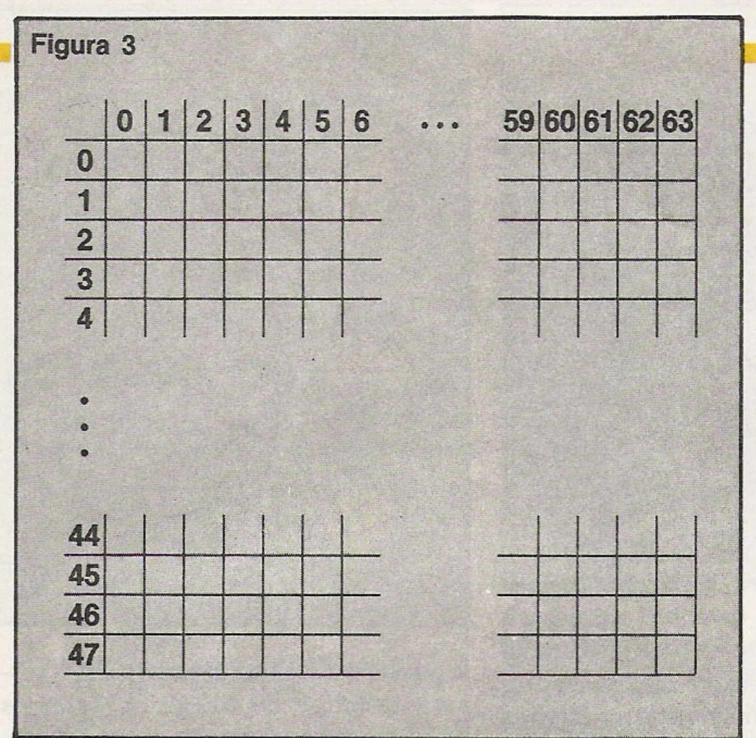


Figura 3

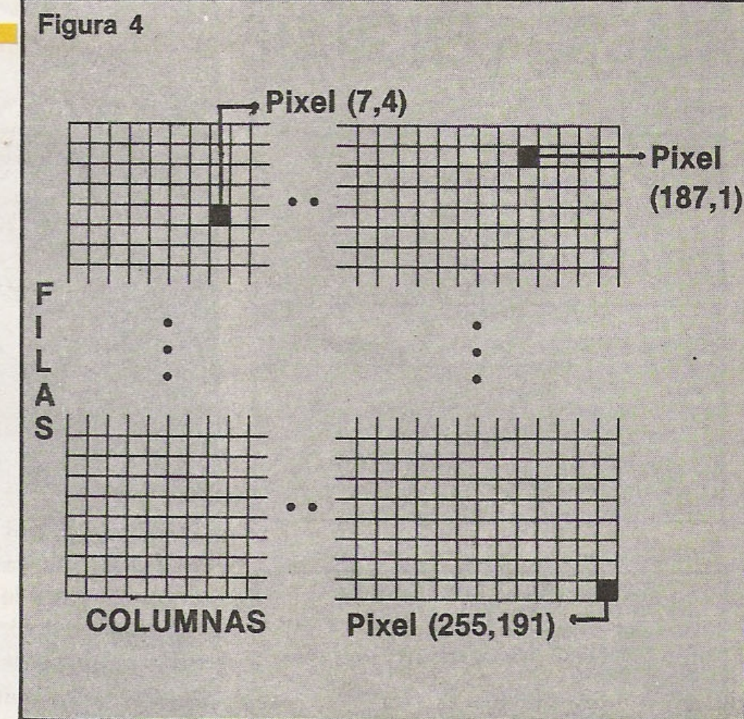


Figura 4

Figura 5

```
10 SCREEN 2: COLOR 1,1,1
16 COLOR 15,1
20 X=INT(RND(1)*14)+1
30 F=INT(RND(1)*254): C=INT(RND(
1)*191)
40 PSET(F,C),X
50 GOTO 20
```

Figura 6

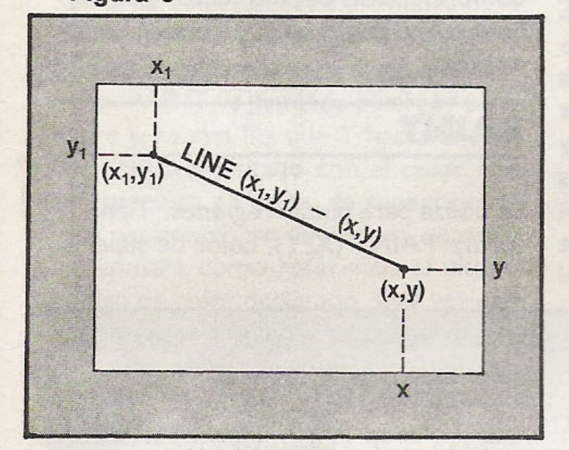
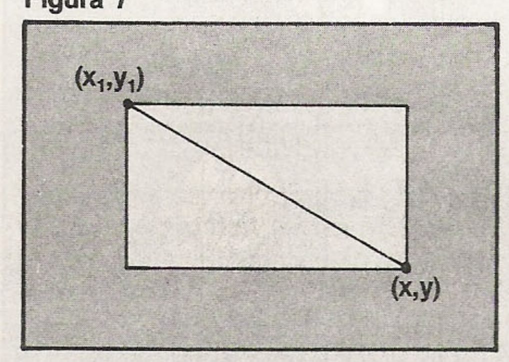


Figura 7



se desplaza hacia la derecha (desde el ángulo superior izquierdo) 150 pixels y a partir de ahí, baja 150 posiciones. Esto es para el primer ejemplo, para el segundo, vuelve a tomar como referencia el cero, cero y se desplaza 2 pixels para la derecha y 4 para abajo.

Si en cambio agregamos la palabra STEP de la siguiente manera: "PSET STEP (X,Y), color" entonces pintaremos el

Figura 8

```
10 SCREEN 2
20 FOR I=1 TO 200 STEP 10
30 FOR C=1 TO 100 STEP 10
40 T=INT(RND(1)*15)
50 T=T+1
60 BEEP
70 LINE(I,0)-(C,190),T
80 NEXT C,I
90 GOTO 90
```

Figura 9

```
10 SCREEN2
20 LINE (61,4)-(150,150):LINE -(
5,60):LINE -(60,4):PAINT (70,70)
30 GOTO 30
```

Figura 10

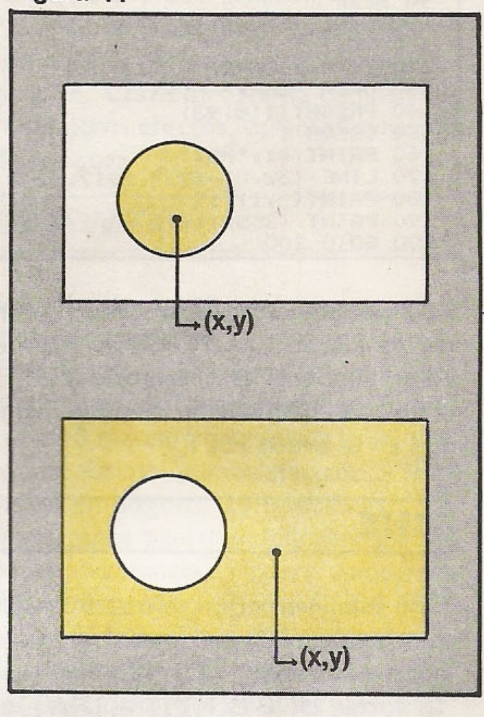
```
10 SCREEN 2:OPEN"GRP:"AS#1
20 FOR I=230 TO 0 STEP -5
30 I=I+1
40 C=INT(RND(1)*15)
50 PSET(100,I+20)
60 LINE-(I-30,100),15
70 PRESET(150,70):PRINT#1,"LOAD"
80 PRESET(60,150):PRINT#1,"MSX"
90 NEXTI
100 FOR I=0 TO 230 STEP 5
110 PSET(100,I+20)
120 LINE-(I-30,100),C
130 NEXT I
140 GOTO 140
```

pixel que se encuentre X lugares horizontales e Y verticales del último pixel ubicado. Por ejemplo PSET (10,10) pina el pixel de la décima columna, décima fila y si a continuación entramos PSET STEP (–3,3), pinta el pixel de la séptima columna, fila 13 del mismo color que el anterior.

Como demostramos con este ejemplo, los valores de X y de Y, pueden ser tanto positivos como negativos.

Se pueden escoger cualquiera de los 16 colores disponibles para pintar los puntos, pero si lo hacemos con el transparente (cuyo código es cero), no podremos visualizar la operación. O sea que el código de color cero simula escribir con tinta invisible.

Figura 11



En la figura cinco, ese listado marcará puntos en cualquier lugar de la pantalla con diferentes colores.

POINT

Con esta instrucción podemos saber de qué color está pintado un pixel determinado. Y se utiliza de la siguiente forma: POINT (X,Y). Veamos un ejemplo: POINT (10,10) que nos devolverá un ocho si este "puntito" estaba pintado de rojo mediano.

Esta sentencia también se puede utilizar con el STEP.

PRESET

Se emplea para cambiar el color del pi-

Figura 12

```
10 SCREEN 2: COLOR 8,1,1
20 CIRCLE (100,100),50
30 PAINT (100,100),8,4
40 GOTO 40
```

Figura 13

```
10 SCREEN 3: COLOR 8,1,1
20 CIRCLE (100,100),50
30 PAINT (100,100),4,8
40 GOTO 40
```

Figura 14

```
10 SCREEN 2:COLOR 11,1,7
20 PAINT (255,96),1
30 DEF FNR(X)=4*ATN(1)
40 FOR T=0 TO 90
50 TIME=0
60 A=T/30*FNR(A)
70 X=128+80*COS(A)
80 Y=96+80*SIN(A)
90 BEEP
100 LINE (128,96)-(X,Y)
110 NEXT T
120 LINE (128,96)-(X,Y)
130 OPEN "GRP:" AS#1
140 PRESET(118,93)
150 COLOR 1
160 PRINT #1,"MSX"
170 LINE (32,0)-(223,191),12,B
180 PAINT(1,1),12
190 PAINT (255,1),12
200 GOTO 200
```

xel especificado en la instrucción. La forma es PRESET (X,Y), color. Aquí el color tampoco es obligatorio y STEP también es utilizable de la misma forma que en la orden PSET.

LINE

Con esta instrucción se trazan líneas o rectángulos. Primero veamos cómo se hacen las líneas.

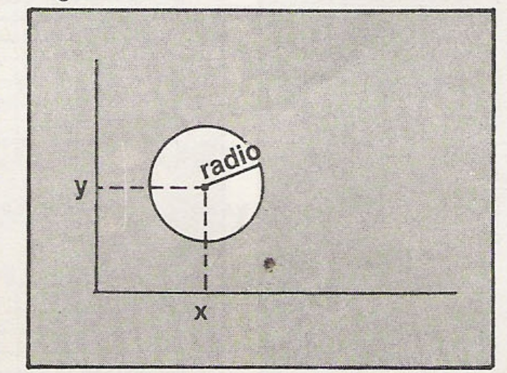
La forma LINE (XI,YI) – (X,Y), color, tipo de dibujo.

El parámetro "tipo de dibujo" puede ser B o BF (más adelante veremos para qué son estos parámetros) y no son obligatorios.

Al obviarse este parámetro, el sistema dibujará líneas. Veamos la figura seis para entender las posiciones que representan X_1, Y_1, X e Y. Como podemos observar, LINE une los puntos (X_1,Y_1) con el (X,Y). Si no le damos valores a los dos primeros, entonces uniremos el último punto al que tuvo acceso el sistema con el punto (X,Y). Para ésto, la orden es LINE - (X,Y). Pero si declaramos como tipo de dibujo: B, entonces dibujará un rectángulo cuya diagonal es la línea que une los puntos (X_1,Y_1) con (X,Y) como se observa en la figura 7.

Pero supongamos que queremos pintar el rectángulo, esto se podría efectuar pintando pixel por pixel, con las instrucciones que vimos antes, pero desde el BASIC esto tardará demasiado. Si en la sentencia LINE le damos el parámetro "tipo de dibujo", el valor BF, en realidad estamos ordenando dibujar un rectángulo (como vimos antes) y además pintarlo.

Figura 15



Al igual que en todas las sentencias para graficar, se puede utilizar el STEP y de la misma forma.

Programemos un ejemplo: como podría ser trazar líneas desde la fila de pixels superior hasta la fila 190. Pero tratemos que las líneas no sean perpendiculares sino oblicuas. Entonces lo graficaremos con LINE (I,0) – (C,190) donde I y C crecen. Para incrementar estas dos variables, podemos emplear dos sentencias FOR, con distintos topes. Como por ejemplo FOR I=0 TO 200 STEP 10 y otra FOR C=0 TO 100 STEP 10. Las dos sentencias incrementarán los valores de I y de C de diez en diez hasta que I alcance el valor 200 y C el 100. Sería interesante pintar cada línea de un color diferente y aleatorio. Sabemos que los códigos de colores varían entre 0 y 15. La función RND(1)*15 genera números al azar entre 0 y 14, en consecuencia si le sumamos 1 al resultado, obtendremos entre 1 y 15 salvando así la posibilidad de escribir con tinta transparente (color de código 0). Pasemos en limpio todas estas sentencias copiando el listado de la figura 8.

Figura 16

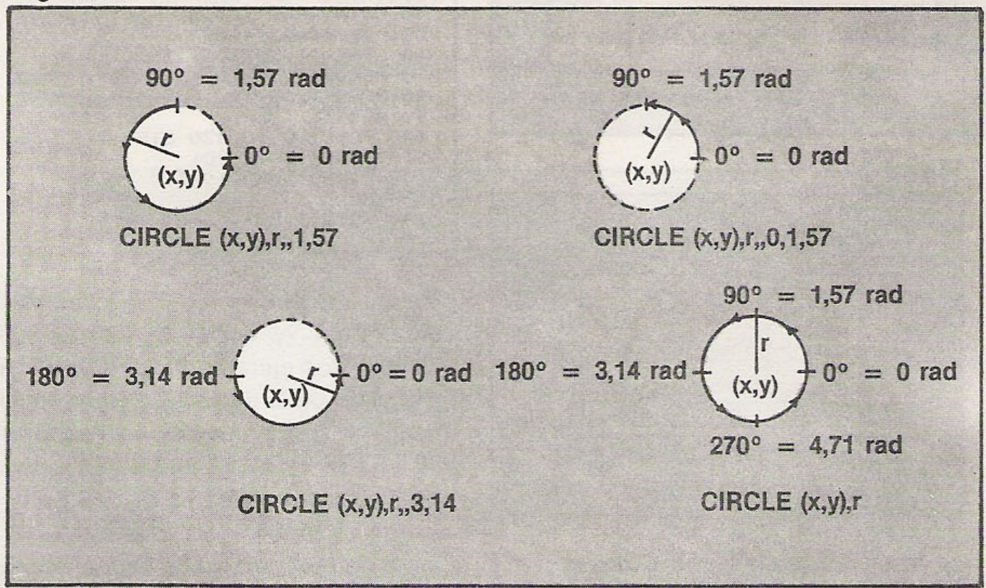


Figura 17

H, A, 2r, r — RAZON = H/A = 2r/r = 2

H, A, 1,5, r r — RAZON = H/A = 1r/1,5r = 0,66

H, A, r — RAZON = H/A = r/r = 1

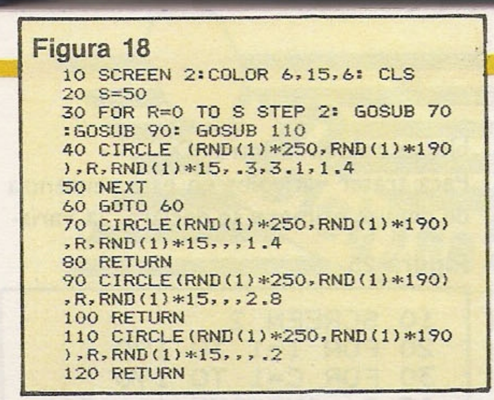
Con LINE también podemos dibujar triángulos como se muestra con el listado de la figura 9.

El listado de la figura 10 es un poco más interesante y aplica todas las instrucciones que vimos hasta ahora. No explicamos este programita porque lo dejamos como ejercicio para el lector pues no hemos ingresado ningún conocimiento nuevo.

PAINT

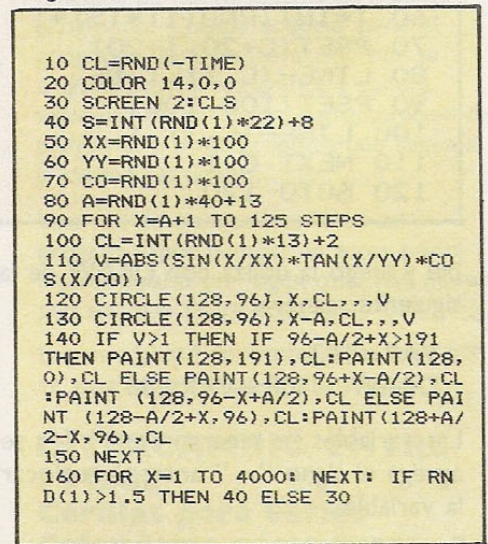
Se utiliza para pintar regiones. Tiene la forma: PAINT (X,Y), color de superficie, color de borde. Hay que indicarle a la computadora qué región queremos colorear y para eso, X e Y es un punto dentro de la zona que queremos pintar. En la figura 11 mostramos que si el punto se halla en el interior del círculo, entonces se pinta la superficie del mismo. En cambio, si el punto es externo a la

Figura 18



```
10 SCREEN 2:COLOR 6,15,6: CLS
20 S=50
30 FOR R=0 TO S STEP 2: GOSUB 70
:GOSUB 90: GOSUB 110
40 CIRCLE (RND(1)*250,RND(1)*190
),R,RND(1)*15,.3,3.1,1.4
50 NEXT
60 GOTO 60
70 CIRCLE(RND(1)*250,RND(1)*190)
,R,RND(1)*15,,,1.4
80 RETURN
90 CIRCLE(RND(1)*250,RND(1)*190)
,R,RND(1)*15,,,2.8
100 RETURN
110 CIRCLE(RND(1)*250,RND(1)*190
),R,RND(1)*15,,,.2
120 RETURN
```

Figura 19



```
10 CL=RND(-TIME)
20 COLOR 14,0,0
30 SCREEN 2:CLS
40 S=INT(RND(1)*22)+8
50 XX=RND(1)*100
60 YY=RND(1)*100
70 CO=RND(1)*100
80 A=RND(1)*40+13
90 FOR X=A+1 TO 125 STEPS
100 CL=INT(RND(1)*13)+2
110 V=ABS(SIN(X/XX)*TAN(X/YY)*CO
S(X/CO))
120 CIRCLE(128,96),X,CL,,,V
130 CIRCLE(128,96),X-A,CL,,,V
140 IF V>1 THEN IF 96-A/2+X>191
THEN PAINT(128,191),CL:PAINT(128,
0),CL ELSE PAINT(128,96+X-A/2),CL
:PAINT (128,96-X+A/2),CL ELSE PAI
NT (128-A/2+X,96),CL:PAINT(128+A/
2-X,96),CL
150 NEXT
160 FOR X=1 TO 4000: NEXT: IF RN
D(1)>1.5 THEN 40 ELSE 30
```

figura, se colorea el complemento de la misma. Si no te interesa saber como trabaja este comando, seguí el siguiente párrafo. PAINT va pintando pixel por pixel sobre la misma fila que Y hasta encontrar un pixel pintado con el color que se especifica en color de superficie. Si este parámetro no se declara, el sistema tomará como referencia al último código de color declarado. Una vez que encontró los límites, se va corriendo hacia arriba y hacia abajo del punto (X,Y), fila por fila. Y en cada fila se vuelven a buscar los límites de la misma forma que explicamos arriba.

Cuando trabajamos con SCREEN 2, alta resolución, el color de borde es ignorado. En cambio en el modo de pantalla 3, se puede pintar las superficies de un color y los bordes de otro. A pesar que los listados de las figuras 12 y 13 son semejantes, los resultados difieren porque un programa trabaja en alta resolución y el otro en baja.

El listado de la figura 14 es un ejemplo de los gráficos que se pueden realizar con la sentencia que acabamos de ver.

CIRCLE

Dibuja círculos, arcos y elipses. Es de la forma CIRCLE (X,Y), radio, color, ángulo inicial, ángulo final, razón axial.

El punto (X,Y) representa al centro de la circunsferencia, veamos la figura 15. El círculo se dibujará en sentido contrario al de las agujas del reloj. El ángulo inicial y final marca los límites del arco a graficar. Pero los ángulos no están expresados en grados sino en radianes. Pero esto no significa demasiadas complicaciones, porque para pasar un ángulo de grados a radianes basta con multiplicar los grados por 3.14 y dividirlo por 180. En la figura 16 marcamos algunos ejemplos.

Faltaría ver que es la razón axial. Esta razón mide cuan achatada o alargada es la elipse con respecto al círculo que se formaría con el radio. Esta razón está dada por la división entre la proporción del radio vertical que queremos achicar o agrandar y el radio horizontal. En la figura 17 mostramos ejemplos.

Esta instrucción también permite usar el STEP y se coloca entre la orden CIRCLE y el punto (X,Y).

Programemos un ejemplo que grafique toda clase de círculos, elipses y arcos al azar en cualquier lugar de la pantalla y con distintos colores. Esto puede parecer complicado, pero es muy sencillo, veámoslo.

La primera instrucción que debemos entrar, es declarar el modo de alta resolución, o sea SCREEN 2. Para obtener cualquier valor de X entre 0 y 250 usemos: RND(1)*250 y para los valores aleatorios de Y: RND(1)*190. Si queremos elegir cualquier color también al azar, lo obtendremos con : RND(1)*15. Y para tener distintos ángulos iniciales y verticales declaramos: RND(1)*6 que generará ángulos en radianes entre 0 y 6, que equivaldría más o menos a ángulos entre 0 y 360 grados.

Pasemos en limpio lo que acabamos de comentar y copiemos el listado de la figura 18. Como programa demostrativo de otros efectos, tipiemos el listado de la figura 19.

Figura 20

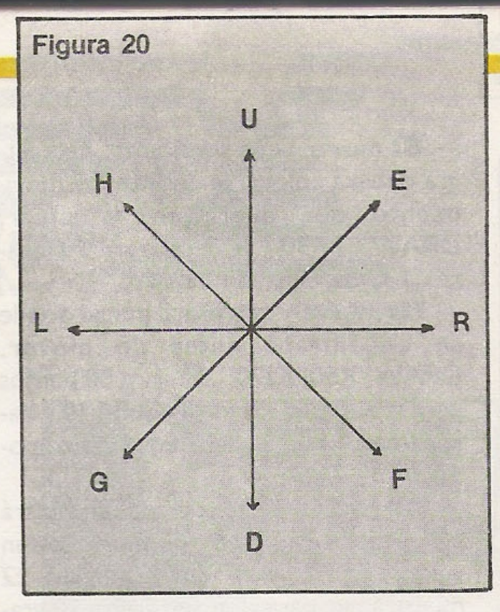


Figura 21

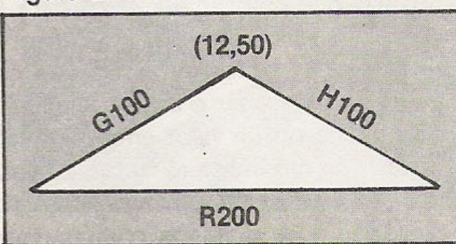
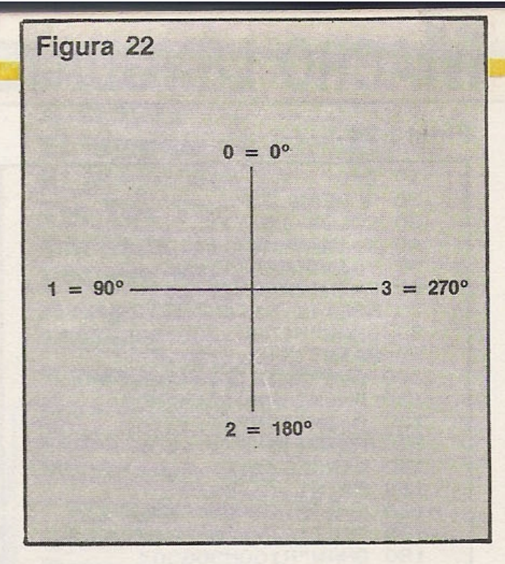


Figura 22



DRAW

Se utiliza para graficar pero emplea un lenguaje particular.

En la figura 20 se puede ver qué letra le corresponde a cada dirección. Entonces U10 marcará diez pixels hacia arriba del último punto dibujado y F20 bajará en diagonal 20 posiciones.

Si queremos dibujar un triángulo aplicando esta sentencia, primero ubiquemos el punto inicial con PRESET (120,50) y color verde mediano, luego bajemos en diagonal hacia la izquierda 100 posiciones con G100, movámosnos hacia la derecha 200 pixels con R200 y por último subamos en diagonal y hacia la izquierda 100 puntitos usando H100. Con estos movimientos hemos dibujado un triángulo (ver figura 21) que como instrucción sería: DRAW "G100R200H100", notemos que los movimientos se colocan entre comillas y todas a continuación.

Pero hay muchas opciones aplicables con el DRAW:

— **M X, Y:** dibuja líneas. Une el último pixel ubicado con el punto (X,Y). Si agregamos un signo "+" o un "–" adelante de los valores de X o de Y, en-

Figura 23

```
10 SCREEN 2: COLOR 2,1,1
20 PRESET(100,100),2
30 A$="C2D50E80H50G140R120F60H15
0"
40 DRAW "XA$;"
50 GOTO 50
```


```
10 SCREEN 2: COLOR 2,1,1
20 PRESET(100,100),2
30 A$="C2D50E80H50G140R120F60H15
0"
40 DRAW "XA$;"
50 GOTO 50
```

Figura 24

```
10 Z=0
20 SCREEN 2
30 COLOR 15,1,1
40 X=INT(RND(1)*128)*2
50 Y=INT(RND(1)*96)*2
60 C=INT(RND(1)*15)
70 PSET(X,Y),C
80 DRAW"U100R120D100L120"
90 PAINT(X+2,Y-2),C
100 O=C-4:IF O<0 THEN 60
110 A=INT(RND(1)*230)
120 B=INT(RND(1)*170)
130 R=INT(RND(1)*50)
140 CIRCLE(A,B),R,O,,,1.5
150 PAINT (A,B),O
160 Q=C-8:IF Q<0 THEN 60
170 PSET(A,Y),Q
180 DRAW"R100H50G50"
190 PAINT(A+25,Y-5),Q
200 Z=Z+1:IF Z<>10 THEN 40
210 GOTO 10
```

tonces se realizarán los movimientos relativos de dichas coordenadas. Por ejemplo, si tenemos M+100,50 se moverá 100 columnas de pixels contando desde la última que se marcó un punto y 50 pixels contando desde la última que se marcó un punto y 50 pixels contando desde el (0,0). Este último punto al no tener signo es contado desde el origen. Si los signos son positivos quiere decir que X e Y aumentan. En cambio si son negativos, disminuyen.

— **Cn:** pintará con el color que se declara con n, siendo n un código de color. C2 pintará en verde claro.

— **B:** mueve pero sin dibujar. Esta letgra deberá colocarse delante del movimiento que querramos efectuar. DRAW "BR50" se desplazará 50 lugares a la derecha sin dibujar.

— **N:** mueve y regresa al punto donde se encontraba antes de mover. DRAW"R50NU20" dibujará 50 puntos para la derecha, desde ahí sube 20 y baja para volverse a ubicar en la misma posición.

— **An:** indica cuantos grados se rotará la figura. En este caso "n" puede ser un valor entre cero y tres. La figura 22 muestra que para girar una figura 90 grados hay que colocar A1, pero para girarla 270 se escribe A3.

— **Sn:** es el factor de escala con que queremos agrandar o achicar el dibujo. "n" puede ser un valor entre 0 y 255. La división n/4 indica el factor de escala. Si n=8 entonces el dibujo será el doble del que se dibujaría comunmente. (8/4=2)

En vez de escribir todas las opciones dentro de la sentencia DRAW, se pueden declarar aparte en una variable alfanumérica y luego agregársela a la orden de dibujar. Esto se demuestra en la figura 23. Si observamos los dos listados notaremos que delante de la variable se coloca la letra "X".

Para tratar variables en esta sentencia de dibujo, primero se declara la variable y luego la utiliza con DRAW de la siguiente manera:

Figura 25

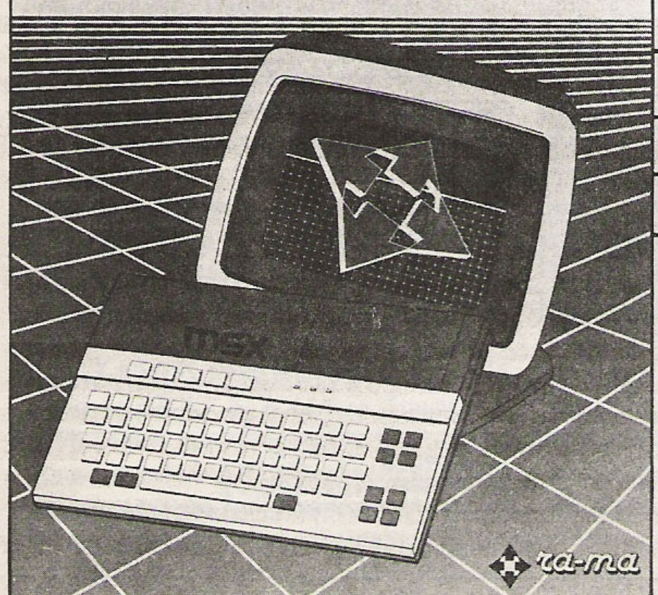
```
10 SCREEN 2
20 FOR I=1 TO 190
30 FOR C=1 TO 190
40 I= INT(RND(1)*190)
50 C=C+1
60 T=INT(RND(1)*15)+1
70 PSET(C+20,I+20)
80 LINE-(C,I),T,BF
90 PSET(100,100)
100 LINE-(C,I),1
110 NEXT C,I
120 GOTO 120
```

VAR=183
DRAW"U50R=VAR;L50"

Las variables se separan con ";" y se agrega el signo "=" antes de colocar la variable.

En el listado 24 usamos todas las opciones que explicamos, y en la figura 25 vemos un ejemplo con muchas de las utilidades que se pueden aplicar para gráficos MSX.

Critica de Libros



MSX
Código máquina.
Programación práctica

Autor:
Steve Webb
Editorial:
RA-MA

Este libro supone que el lector no posee conocimientos previos sobre el código de máquina. Comienza explicando las equivalencias en código máquina de las principales instrucciones BASIC, tales como IF, FOR/NEXT, PRINT, GOTO, GOSUB, etc...

Continúa con la explicación detallada de las rutinas individuales de un simple juego: "INVASORES DEL ESPACIO", y cómo se enlazan éstas para aprovecharlas en programas BASIC.

También hay, a lo largo del libro, numerosas preguntas que autoevaluan nuestros conocimientos sobre la programación en lenguaje Assembler.

Se trata de una informal y divertida forma de aprender un tema que puede resultar escabroso y decepcionante. Es una buena opción para aquellos que nunca hayan tenido relaciones con este asombroso lenguaje.



Dentro del Club de Usuarios MSX ya funciona un nuevo sistema para adquirir programas. Cuando el socio envíe el programa, éste será evaluado por su originalidad, calidad gráfica y sonora, velocidad de ejecución, estructuración del programa y otros atributos que hacen a la calidad del diseño.

Una vez determinado el valor del programa (por puntos), el autor podrá intercambiarlo por algun otro (del mismo puntaje) que figure en la lista.

Esta modalidad de intercambio, hará que se vaya incrementando dicha lista.

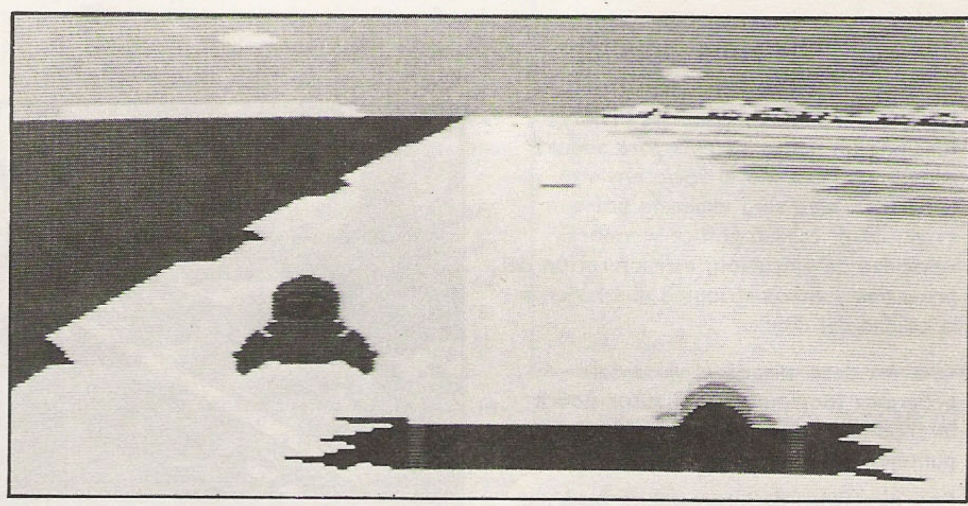


INTERCAMBIO ENTRE SOCIOS

Cerillas para varios
Calendario perpetuo
Batalla naval
Golf
Solfeo
El juego del 21
Par - Impar
Adivinar una cifra
El juego de witthof
Rebotes
Jackpot
Ruleta
Pasillos
El juego del haba
Juego de la oca aleatorio
Dimensionamiento de una tabla
Cadena
El electricista loco
La carrera al 20
Musica
Poker
El juego de la pidola
Letras
Máquina de palabras
Espia
Juego de la loteria
Tirada de un dado trucado
El juego de marienbad
Componer
Día de la semana
Juego de cartas
El indeseable
El juego de la vida
El ahorcado
Belote
Un cálculo fastidioso
Ahorcado al reves
Nicomaque
Palabras
Permutaciones
La diana de Robin Hood
Adivina el Número
El gorila matemático
Minotauro Mastermind
Minotauro Mastermind Jr.
Diseños
Cocodrilo
Convoy
Dragster
Batalla laser
Invasión de robots
O.V.N.I.
Ataque astral
Las cucarachas cosmicas
Concentración
Test de inteligencia
En busca del tesoro
Treinta y una
Acrobata aereo
El saltarín del río
Emboscada de tanques
Bridge
Hanoi
Carrera de caballos
Enganchar los vagones
El lobo y los corderos
Combate aereo
Caza del submarino
Eljuego del misil
El mono
Destreza
Submarino
Zombies
Calculadora
Pizarra mágica
Evasión
Reloj 2
Repeticiones
A través
El lagarto de Alicia
El juego del 15
Tic Tac Toc
Carrera de coches
Peligro espacial
Anagramas
Huevos
Revoltijo de letras
Obstáculos
Capturas en el espacio
Varios en uno
Pared de ladrillos
Othello
Damas
De cuatro en cuatro
Ajedrez
La Gallina ciega
Codificador de resistencias
Satélite espacial
Salta cubos
Gráficos aleatorios
Lapiz msx
Caracteres especiales
Gestión de datos y de discos de larga duración
Calendario
Diagrama de barras
Tablas deportivas
Hollow - el juego de las cerezas
Reacción
Descifrador de códigos
Espanta intrusos
Hagamos música
Objetivo: Nueva York
Indianapolis
Breakout
El archivo en casa
La entrella de la Muerte
Alienigenas del espacio
Como diseñar un sprite
Solfeo
Cross
Curso de mecanografia
Rescate aereo
Caseta de tiro
Globos
Harrier
Naves secretas de investigación
Planetoide
Prueba de futbol
Editor gráfico
Editor de sonido
Graficos
Tron
El borra
Matemáticas
Poker
Diseñador de sprites
Artista
Cavernous
Pareja



AVENTURE ANTARTIC

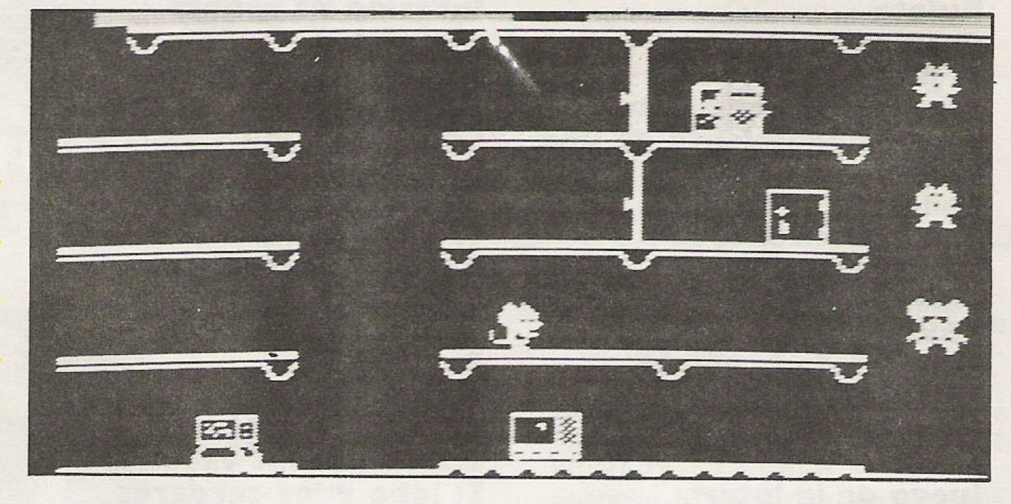


CREATIVIDAD: 9
PRESENTACION: 6
ATRACCION: 9
GRAFICOS: 8
SONIDO: 9
TIPO: ENTRET.
PRODUCE: BITGAME

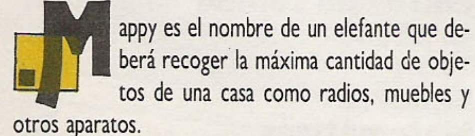
Se trata de un pingüinito muy simpático que recorre la Antártida. Durante el trayecto, tiene que recoger la mayor cantidad de peces que sea posible para poder alimentarse.
Tiene un tiempo limiado para recorrer el camino entre estación y estación de este continente frío. Cada estación pertenece a una colonia antártica diferente.

Si cae en un pozo perderá tiempo y lo mismo sucede si choca contra una foca.
Este entretenimiento sale de lo común porque nadie come a nadie ni hay matanzas como suele suceder en la mayoría de los juegos.
Es un pasatiempo inofensivo y tierno.
Si queremos tomar velocidad, hay que avanzar, pero caer en los pozos implicará disminuir la rapidez.
Este simpático pinguino, al llegar a la estación antártica, nos saludará y seguirá el recorrido señalado en el mapa que aparece al comienzo de juego.
Es muy difícil que nos cansemos de usar este programa porque tiene varios niveles y seguramente nadie parará hasta saber que nos encontraremos en el último.
Para dificultarnos las cosas, a medida que pasamos estaciones, el tiempo para jugar es menor.
Tiene una pequeña demostración que nos facilitará entender como se juega.
El programa nos permite usar las teclas de cursor o los joysticks.
Estan cuidados todos los detalles como bigotes en las focas, peces saltarines con movimiento, nubes, y lo más atractivo, una música muy parecida a la de los organitos antiguos.
Pero no hemos podido averiguar que busca este simpático personaje por la Antartida.

MAPPY



CREATIVIDAD: 8
PRESENTACION: 7
ATRACCION: 8
GRAFICOS: 9
SONIDO: 8
TIPO: ENTRET.
PRODUCE: BITGAME



Mappy es el nombre de un elefante que deberá recoger la máxima cantidad de objetos de una casa como radios, muebles y otros aparatos.
Hay puertas que las tendremos que abrir para poder pasar, eso lo conseguimos presionando la barra de espacio o el botón de disparo del joystick, depende con que juguemos.
Hay algunas puertas que al abrirlas, el elefante grita y con eso mata a los ratones.

Las sogas haran que subamos pisos ya que a la casa se olvidaron de fabricarle escaleras.
Pero si saltamos más de tres veces seguidas sobre estas sogas, se romperan y caeremos perdiendo una vida.
Hay muchos ratones que perseguiran a Mappy, si consiguen alcanzarlo, Mappy morirá de susto porque les tiene pánico a los ratones.
Entre todos los ratones hay mezclado también un gato, que si se encuentra en alguno de los objetos que tiene que recoger el elefante, será inofensivo. Entonces Mappy podrá tomar dicho objeto y sumar más puntos.
En el nivel siete, aparecerán muchos ratones y todos son muy veloces, más que el elefantito. En el siguiente nivel, el piso se rompe por secciones y esto producirá graves consecuencias porque no nos permitirá volver a pasar por esas zonas.
La barra de los costados nos protegen de salir de la casa.
Si llegaramos a saltar cuando no estan dichas barras, perderemos una vida.
Cuando los objetos titilan, es la oportunidad de juntar más cantidad de puntos.
El diseño de cada nivel es variado, y la música de fondo es tan entretenida que pasaremos varias horas frente a este juego antes de abandonarlo.

STAR AVENGER

CREATIVIDAD: 5
PRESENTACION: 6
ATRACCION: 7
GRAFICOS: 7
SONIDO: 6
TIPO:ENTRET.
PRODUCE: BITGAME





Este juego está dirigido para aquellos jugadores que le gusten las aventuras espaciales agresivas.

Se trata de una nave que debe destruir todo tipo de objetos extraños como misiles, bases enemigas, ramificaciones de frutas, vampiros extraterrestes y bolas de fuego.

También hay tanques de gas oil y al destruirlos, abasteceremos a nuestra nave.

La batalla tiene varios escenarios que puden ser en planetas, tuneles, cuidades, cavernas, laberintos, o con aeroplanos enemigos que nos dispararan.

Los gráficos no son demasiado originales, pero son agradables.

Tiene algunos efectos que nos marearan y no sabremos por donde anda nuestra nave.

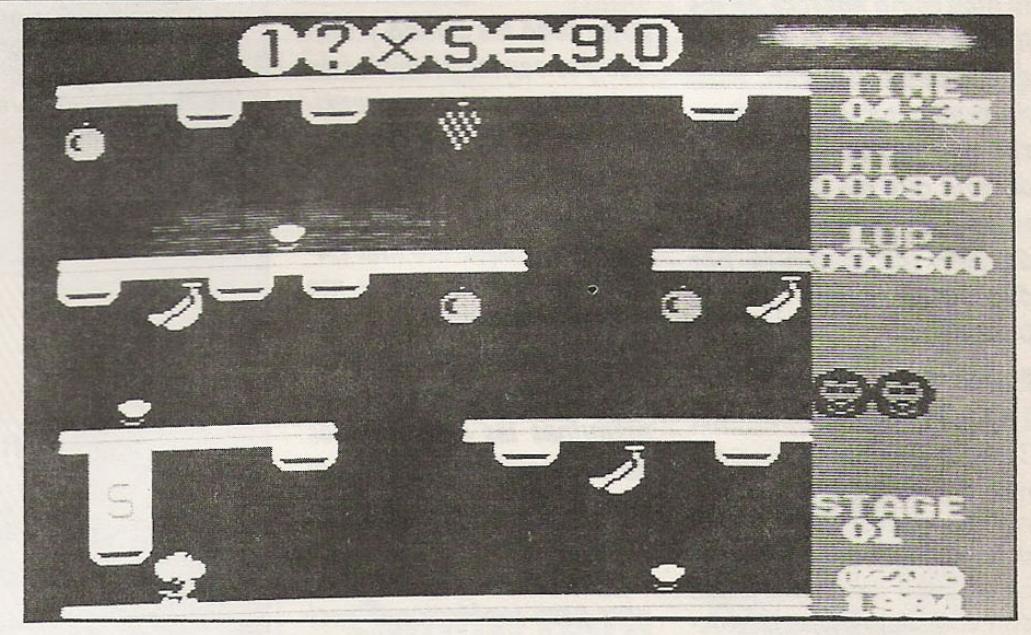
El sonido se escucha durante el juego, pero tiene una opción que permite eliminarlo. Si presionamos "P", produciremos una pausa durante la ejecución del programa. La nave se maneja con elteclado, y no tiene la opción joystick.

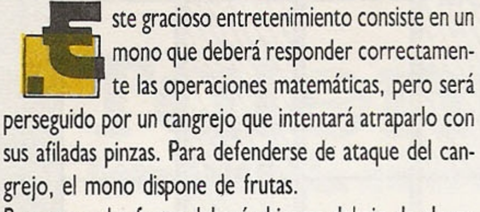
La tecla "R" mueve la nave hacia arriba, "V" la baja, "H" retrocede la nave y "K" la adelanta.

Pero insistimos que se trata de un entretenimiento muy agresivo, que poco sale de lo común de los juegos de masacres espaciales.

MONKEY ACADEMY

CREATIVIDAD: 9
PRESENTACION: 6
ATRACCION: 7
GRAFICOS: 9
SONIDO: 8
TIPO: EDUCATIVO
PRODUCE: BITGAME





Este gracioso entretenimiento consiste en un mono que deberá responder correctamente las operaciones matemáticas, pero será perseguido por un cangrejo que intentará atraparlo con sus afiladas pinzas. Para defenderse de ataque del cangrejo, el mono dispone de frutas.

Para tomar las frutas deberá ubicarse debajo de alguna y saltar, lo mismo se hace si quiere bajr las cortinas para ver si el resultado que busca se encuentra en ese lugar.

Para disparar la fruta contra el perseguidor, también se deberá saltar y de esa forma la fruta que hayamos recolectado será lanzada.

Una vez que hayamos encontrado la cortina que tiene el resultado, presionamos la tecla de SELECT y caerá un lápiz. Este debe ser llevado al mono que aguarda impaciente en la parte superior de la pantalla, para que lo ubique en el casillero correcto.

Hay varios niveles de dificultad, y en cada uno varía el escenario siendo en algunos casos bastante difícil llegar hasta el otro mono que nos espera.

En el nivel uno y dos, las operaciones son relativamente sencillas, mientras que en los niveles superiores las cuentas se complican hasta tener que resolver ecuaciones con paréntesis.

Este juego es ideal para poder adquirir rapidez en cálculos mentales.

Durante todo el juego una música suave y entretenida nos acompañará.

No se aceptará que erremos más de tres veces las respuestas de las cuentas, si esto sucede, perderemos el turno. Y al llegar a los 30.000, obtendremos una vida extra.

Para subir pisos, hay que tomar envión, y si conseguimos tener en la mano el lápiz, no podremos recoger frutas.

Este es uno de los entretenimientos que mientras nos permite jugar, no nos hacen sentir que desperdiciamos demasiado el tiempo.

Obervemos bien las cantidades y tengamos a mano papel y lápiz, porque algunas cuentas no son tan fáciles.



Para comunicarse con nosotros deben escribirnos a Revista para usuarios de MSX, Paraná 720, 5to. Piso, (1017) Cap. Federal.

UBICACION DEL CURSOR

En algunos programas, al estar trabajando con gráficos, traté varias veces de ubicar el cursor con LOCATE, en algún lugar determinado de la pantalla y no lo he conseguido. Por esto, dejo que los mensajes salgan en cualquier lugar de la pantalla, según sea el antojo de la computadora. Y lo más extraño es que con otras instrucciones como PSET, la siguiente impresión será a partir de ese punto.
No me digan que abro mal el archivo de pantalla, porque lo hice como aparece en los listados de los programas que ustedes publican. Pero no puedo agregar el capricho que les conté. Les agradecería que me den alguna sugerencia porque me cansé de leer los mensajes por cualquier lugar de la pantalla.

Martín Soto, Capital

LOAD MSX

Tu problema es bastante común a todos los programadores que comienzan con este sistema standard. Pero todo es cuestión de familiarizarse con el manejo de las instrucciones. Hay algunas diferencias que deberás tener presente cada vez que realices un programa. Como te habrás dado cuenta hay modos de texto y modos gráficos. Algunas instrucciones solo se pueden ejecutar para uno de esos dos tipos diferentes de pantalla. Por ejemplo, es imposible ejecutar una instrucción de dibujo en los modos de texto (0 o 1). Lo mismo sucede si intentamos imprimir directamente cuando trabajamos con SCREEN 2.
La instrucción que intentaste usar en el modo gráfico, no es aplicable para éste. Si querés ubicar el cursor, te aconsejamos hacerlo con PRESET para cualquiera de los gráficos dos o tres. Habrás notado que al hacerlo con PSET, te pinta un pixel, esto se evita con la instrucción que te recomendamos. Y no te aconsejamos rendirte ante los caprichos de la computadora, consultá tus dudas y juntos las solucionaremos.

IMPRESORA PARA MSX

Ante todo los saludo y felicito por ser los primeros en guiar a los usuarios argentinos de MSX y les deseo el mayor de los éxitos. Soy un principiante de MSX ya que recién he adquirido una TALENT MSX DPC 200 y me gustaría saber:
**** ¿Cuándo estarán a disposición de los usuarios de estas máquinas las ampliaciones de memoria? ya que según el manual y los folletos es extendible a 576K.***
**** ¿Qué impresora podría usar para confeccionar recibos de sueldo, listados de stock, facturas, etc.?, ya que la impresora TALENT es demasiado pequeña para lo que necesito.***

Jorge Caldoli - 25 de Mayo 426, Bernal

LOAD MSX

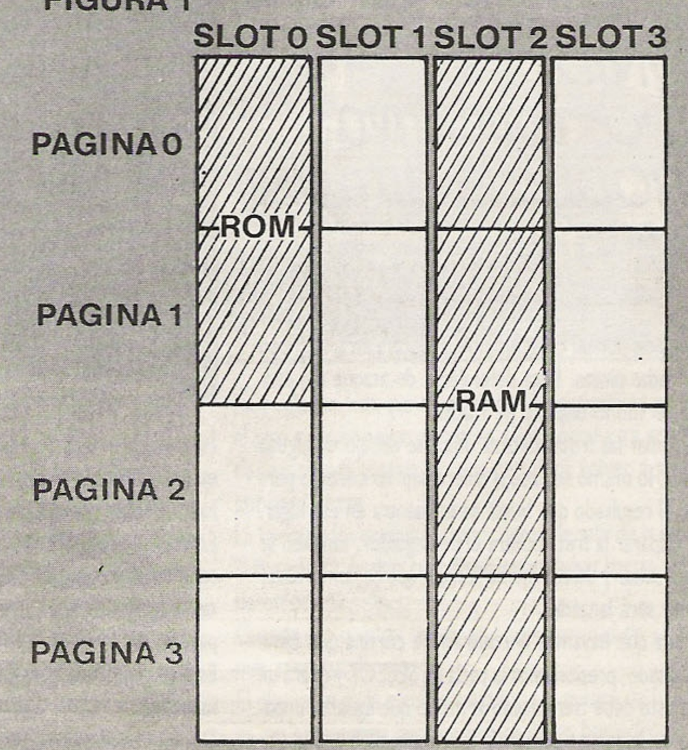
Sería muy interesante disponer de los cartuchos expansores de memoria que mencionaste, nosotros también los esperamos. Pero eso depende de los fabricantes de HARDWARE como TALENT, que todavía no nos han confirmado nada sobre este tema.
En relación a tu segunda pregunta, la impresora de TALENT es más que eso, es un ploter. Esto quiere decir que esta impresora te permitirá graficar círculos, rectángulos o cualquier figura que desees. Es cierto que es pequeña, pero si no querés dibujar con la impresora, no es necesario alguna máquina especial. En tonces cualquier impresora con salida paralela te sirve.

PERDIDA DE MEMORIA

Tengo una TALENT MSX y mi problema es que solamente puedo encontrar 28 K libres para BASIC. ¿Podrían ustedes decirme dónde están los restantes KBYTES y cómo se puede acceder a ellos?
César Alcaraz, Capital

LOAD MSX

Los microprocesadores de 16 bits de direcciones, como usa la Talent MSX, pueden acceder a un máximo de 64 K directamente.
En el apéndice C del manual de la computadora, encontramos el mapa de memoria. (Ver figura 1).
Las páginas cero y uno del slot 0 es ocupado por ROM. En el slot 2, hay 64 K de RAM disponibles, pero al encender la computadora, parte de esta memoria no es accesible porque el sistema las reserva para operaciones.
Esto deja sólo los 28 K de RAM libres para el BASIC. Esta cantidad de memoria libre es la misma para las máquinas con 32K que para las de 64K.
La ventaja de tener mayor capacidad de memoria se nota cuando se quiere utilizar el código de máquina, porque esto puede ir en el área ocupada normalmente por el BASIC. En consecuencia, al trabajar con Assembler, contamos con los 64K disponibles para nuestros programas.



Computación, una oportunidad para que todos enseñen y aprendan.

Un lugar para

desarrollar el pensamiento.

descubrir una vocación.

manejar lenguajes de computación.

comprender los múltiples usos de un computador.

capacitar y perfeccionar al docente.

incorporar los avances tecnológicos.

que el profesional domine el uso de nuevas herramientas.

que los padres se reencuentren con sus hijos.

"No se trata solamente de adquirir en forma puntual conocimientos definitivos, sino prepararse a elaborar a lo largo de toda la vida, un saber en constante evolución y de aprender a ser."

UNESCO

Actividades '86

Para Niños, Adolescentes, Adultos, Docentes, Profesionales y Establecimientos educativos.

INTRODUCCION A MICROCOMPUTADORES

DIAGRAMACION ESTRUCTURADA

LOGO

BASIC

COLOR - SPRITE - SONIDO

COBOL

PASCAL

ASSEMBLER

MS - DOS Y MSX - DOS

D BASE II - MULTIPLAN

PROCESADOR DE LA PALABRA

INSTALACION DE LABORATORIOS

en Establecimientos educativos con formación de multiplicadores y apoyo a la comunidad.

Cómo?

- Taller en grupos de 12 a 15 personas.
- Clases de 2 horas diarias.
- 2 ó 3 alumnos por equipo.
- Equipos disponibles para prácticas adicionales en horarios libres.
- Becas rentadas en el Departamento de investigación y desarrollo de Talent MSX.
- Becas rentadas para docentes en Laboratorios de Establecimientos Educativos.

Informes, Inscripción y Cursos

Lunes a Viernes de 8 a 22 hs.
Sábados de 8 a 13 hs.
CENTRAL:
Cabildo 2027 - 1er. Piso "A" (1428)
Capital Federal
FILIALES:
Centro: Esmeralda 320 - 3° (1343)
Capital Federal
Lanús: Caaguazú 2186
L. Este
Capital: Tucumán 2044 - 1° (1050) Capital Federal

Talent MSX
Inteligencia en crecimiento.

Centro para el desarrollo de la inteligencia.

Descubramos y construyamos juntos los caminos que nos permitirán el uso inteligente de los productos de la creatividad humana.

diálogo - *D.P.*

Titulos no oficiales

A la computadora personal Talent MSX nada le es imposible

dialogo - D.P.



MSX es marca registrada de MICROSOFT CORPORATION

Porque gracias a la norma internacional MSX, la TALENT MSX trasciende sus propios límites.
Hasta ahora, cuando usted compraba una computadora personal de cualquier marca, quedaba automáticamente desconectado del resto del mundo de la computación. Porque los distintos equipos y sistemas no eran compatibles entre sí.
Hasta que dos grandes empresas de informática, la Microsoft Corp. de EE.UU. y la ASCII del Japón se pusieron de acuerdo para crear una norma standard: la MSX. Que se expandió también rápidamente en Europa. Y que hoy TALENT presenta por primera vez en la Argentina.
Mientras que la mayoría de las computadoras de su tipo que se ofrecen en el mercado nacional, han sido discontinuadas por obsoletas en sus lugares de origen, TALENT MSX tiene casi ilimitadas posibilidades de desarrollo. Porque la norma MSX es en todo el mundo inteligencia en crecimiento.
La TALENT MSX pone a su disposición un mundo de software para elegir. Y con la incorporación de todos sus periféricos llega a ser una auténtica computadora profesional.

UTIL

Su poderoso sistema operativo MSX permite el acceso a todo tipo de procesamiento de datos:

- Planillas de calculo.
- Procesadores de palabra.
- Graficos de negocios.
- Bases de datos (d Base II, etc.)
- Contabilidad general, sueldos, y jornales, costos, etc., desarrollados bajo CP/M en Basic, Cobol, Pascal o C.

Con la posibilidad de conexion a linea telefonica permite la transferencia y consulta de datos entre computadoras personales, profesionales o bancos de datos.
La grabacion de archivos es en formato MS-DOS, haciendola compatible con las computadoras profesionales.

DIDACTICA

Dispone de tres lenguajes para la enseñanza de computacion: LOGO como lenguaje de induccion para los mas chicos. Lenguaje de Programacion en castellano, para todos los que quieran aprender a programar sin conocimientos previos. Y Basic MSX como lenguaje profesional. Mas una amplia variedad de perifericos como el Mouse, Lapiz Optico, Tableta grafica, Track-ball, etc.

DIVERTIDA

La mas genial para Video-Juegos. Por la amplisima biblioteca de programas **–todos nuevos–** de la norma MSX en el mundo. Y ademas, el Basic MSX permite al usuario generar sus propios juegos con un manejo tan simple, como solo TALENT MSX puede ofrecer.

CARACTERISTICAS TECNICAS

- Memoria principal 64 KB ampliable hasta 576 KB.
- Memoria de video: 16 KB RAM.
- ROM incorporada de 32 KB con el MSX-Basic de Microsoft.
- Graficos completos, hasta 32 sprites y 16 colores simultaneos.
- Generador de sonido de 3 voces y 8 octavas.
- Conexion para cualquier grabador.
- Interfaz para salida impresora paralela.
- Conectores para cartuchos y expansiones.
- Fuente para 220 V y modulador PAL-N incorporado.

DISTRIBUIDORES AUTORIZADOS: CAPITAL FEDERAL: AMATRIX, Bolívar 173 - ARGECINT, Av. de Mayo 1402 - BAIDAT COMPUTACION, Juramento 2349 - COMPUPRANDO, Av. de Mayo 965 - COMPUSHOP, Córdoba 1464 - COMPUTIQUE, Córdoba 1111, E. P. - COMPUTRONIC, Viamonte 2096 - CP67 CLUB, Florida 683, L. 18 - DALTON COMPUTACION, Cabildo 2283 - ELAB, Cabildo 730 - MICROSTAR, Callao 462 - Q.S.P., Bartolomé Mitre 864 - SERVICIOS EN INFORMATICA, Paraná 164 - DISTRIBUIDORA CONCALES, Tucumán 1458 - MICROMATICA, Av. Pueyrredón 1135 - **ACASSUSO:** MICROSTAR ACASSUSO, Eduardo Costa 892 - **AVELLANEDA:** ARGOS, Av. Mitre 1755 - **BOULOGNE:** COMPUTIQUE CARREFOUR, Bernardo de Irigoyen 2647 - **CASTELAR:** HOT BIT COMPUTACION, Carlos Casares 997 - **LANUS:** COMPUTACION LANUS, Caaguazú 2186 - **LOMAS DE ZAMORA:** ARGESIS COMPUTACION, Av. Meeks 269 - **MARTINEZ:** VIDEO BYTE, Hipólito Yrigoyen 32 - **RAMOS MEJIA:** MANIAC COMPUTACION, Rivadavia 13734 - **SAN ISIDRO:** FERNANDO CORATELLA, Cosme Beccar 249 - **VICENTE LOPEZ:** SERVICIOS EN INFORMATICA, Av. del Libertador 882 - **BAHIA BLANCA:** SERCOM, Donado 327 - SUMASUR, Alsina 236 - **LA PLATA:** CADEMA, Calle 7 Nº 1240 - CERO-UNO INFORMATICA, Calle 48 Nº 529 - **MAR DEL PLATA:** FAST, Catamarca 1755 - **NECOCHEA:** CAFAL, Calle 57 Nº 2920 - SERCOM, Calle 57 Nº 2216 - **TRENQUE LAUQUEN:** COMPUQUEN, Villegas 231 - **CORDOBA:** AUTODATA, Pasaje Santa Catalina 27 - TECSIEM, Santa Rosa 715 - **ROSARIO:** 2001 COMPUTACION, Santa Fe 1468 - MINICOMP, Maipú 862 - SISOR, Urquiza 1062 - **SANTA FE:** ARGECINT, P. San Martín 2433, L. 36 -SISOR, Rivadavia 2553- INFORMATICA, San Gerónimo 2721/25 - **VILLA MARIA:** JUAN CARLOS TRENTO, 9 de Julio 80 - **MENDOZA:** INTERFACE, Sarmiento 98 - BIT & BYTE, 9 de Julio 1030 - **COMODORO RIVADAVIA:** COMPUSER, 25 de Mayo 827 - **GENERAL ROCA:** DISTRIBUIDORA VECCHI, 25 de Mayo 762 - **LA PAMPA:** MARINELLI, Pellegrini 155 - **NEUQUEN:** MEGA, Perito Moreno 383 - EDISA, Roca esq. Fotheringham - **RIO GRANDE:** INFORMATICA M & B, Perito Moreno 290 - **SAN CARLOS DE BARILOCHE:** L. ROBLEDO & ASOCIADOS, Elfein 13, Piso 1º - **TRELEW:** SISTENOVA, Sarmiento 456 - **PARANA:** MARIO GARCIA, Laprida y Santa Fe - **POSADAS:** CENTRO DE COMPUTOS ELDORADO, Colón 2429 - **RESISTENCIA:** FRANCO SANTI, Carlos Pellegrini 761 - **SAN SALVADOR DE JUJUY:** DELTA COMPUTACION, Salta 1108 - **SALTA:** DELTA COMPUTACION, Caseros 873 - **SAN MIGUEL DE TUCUMAN:** LEXICON, 9 de Julio 85 - ELECTRONICA